AN ARCHITECTURE FOR IMPROVING TIMELINESS AND RELEVANCE OF CYBER INCIDENT NOTIFICATIONS

THESIS

James L. Miller
Master Sergeant, USAF

AFIT/GCO/ENG/11-09

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT/GCO/ENG/11-09

AN ARCHITECTURE FOR IMPROVING TIMELINESS AND RELEVANCE OF CYBER INCIDENT NOTIFICATIONS

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science

James L. Miller, BS
Master Sergeant, USAF

March 2011

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT/GCO/ENG/11-09

AN ARCHITECTURE FOR IMPROVING TIMELINESS AND RELEVANCE OF CYBER INCIDENT NOTIFICATIONS

James L. Miller, BS
Master Sergeant, USAF

Approved:

//SIGNED// 02 March 2011

______________________________ ______________

Dr. Robert F. Mills, (Chairman) (Date)

//SIGNED// 04 March 2011

______________________________ ______________

Dr. Michael R. Grimaila, (Member) (Date)

//SIGNED// 04 March 2011

______________________________ ______________

Dr. Michael W. Haas, (Member) (Date)

AFIT/GCO/ENG/11-09

Abstract

This research proposes a communications architecture to deliver timely and relevant cyber incident notifications to dependent mission stakeholders. This architecture, modeled in Unified Modeling Language (UML), eschews the traditional method of pushing notifications via message as dictated in Air Force Instruction 33-138. It instead shifts to a “pull” or “publish and subscribe” method of making notifications. Shifting this paradigm improves the notification process by empowering mission owners to identify those resources on which they depend for mission accomplishment, provides a direct conduit between providing and dependent mission owners for notifications when an incident occurs, and provides a shared representation for all with authority for that dependent mission. Once the incident’s impact is assessed, the architecture provides a conduit for the mission stakeholder(s) receiving the incident notification to then notify their downstream users of their status should it have changed because of the incident. The proposed architecture significantly speeds incident notification by eliminating multiple layers of processing and does so in a relatively noise-free environment as compared to current notification methods.

To the men and women of
Sierra Pete, Bigfoot, and Loki…

Acknowledgements

I would like to express my sincerest thanks to Dr. Robert Mills, Dr. Michael Grimaila, and Dr. Michael Haas. Your collective enthusiasm for the Cyber Incident Mission Impact Assessment project was infectious and the space available here does not allow me to fully document my appreciation for the patience, mentorship, and scholarly advice that each of you have extended to me during the course of this journey.

I would also like to thank my enlisted wingmen who maintained a vigilant watch of "my six" over what has been a challenging twenty-one months for all of us. Safe travels and fair winds to each of you.

Last and 180º from least, I offer my heartfelt thanks to my bride and my three children for the latitude and airspace provided for the duration of this flight. For this I can offer only my gratitude and unconditional love.

James L. Miller

Table of Contents

Abstract iv

Acknowledgements vi

List of Figures x

List of Tables xi

I. Introduction 1

1.1 Background 1

1.2 Problem statement 2

1.3 Research goals 2

1.4 Scope, Assumptions, and Limitations 3

1.5 Methodology 5

1.6 Preview 5

II. Background Information 6

2.1 CIMIA – Definition, Description, and Components 6

2.1.1 Cyber Incidents 7

2.1.2 Mission, Mission Capability, and Impact Assessment 10

2.2 Communication Issues Relevant to Notifications 14

2.2.1 Identifying Critically Downstream Consumers 15

2.2.2. Failures in Internal Organization and Communication 17

2.2.2.1 Communication Impairment Caused By Organizational Structure 17

2.2.2.2 Communication Impairment from Mission Execution Ignorance 24

2.2.3 Lack of an Automatic Notification System 25

2.2.4 The Dynamic Nature of Organizations and Their Missions 26

2.2.5 The Determination of Criticality is Organization Dependent 27

2.3 Situational Awareness (SA) 27

2.4 Workflow and Business Process Modeling 29

2.5 Pushing versus Pulling 30

III. Investigative Questions 33

3.1 Research Problem: Can timely and relevant incident notifications be made to enduring mission stakeholders? 33

3.2 Primary Investigative Question 1: In accordance with AFI 33-138, C4 NOTAMs are the means in which incident notifications are transmitted. What is keeping C4 NOTAMs from consistently being timely and relevant? 34

3.2.1 Investigative Question: How would pull improve the notification process? 40

3.2.2 Investigative Question: What would such an architecture look like to support the "publish and subscribe" function? 41

3.3 Primary Investigative Question 2: How would this architecture support the receipt of mission-relevant notifications? 46

3.3.1 Investigative Question: How does the mission-level agent pull statuses? 53

3.3.2 Investigative Question: How does the user-level agent become relevant to enduring missions? 60

3.3.3 Primary Investigative Question 3: What details are missing? 62

IV. Methodology 64

4.1. Overriding considerations 64

4.2 Pull 65

4.3 Scalability 65

4.4 Workflow modeling decision 66

4.5 UML and Systems Analysis 68

V. Use Cases, Structural Models, and Behavioral Models 71

5.1 Purpose and Introduction 71

5.4. Overall Use Case Diagram and Class Diagram 80

5.5 Use Case 1: Create a Registered User 86

5.6 Use Case 2: New Mission Status File (MSF) or Asset Status File (ASF) is Created 91

5.7 Use Case 3: Authority for the Mission or Asset is Delegated or Shared 96

5.8. Use Case 4: Add a Mission or Asset Dependency 99

5.9. Use Case 5: Update an MSF or ASF Status 104

5.10 Use Case 6: Monitored Status File Receives an Alert 110

5.9 Conclusion 119

VI – Case for Action 120

6.1 Introduction 120

6.2 Current Notifications Methods are Insufficient 120

6.3 Threat to Our Networks Continues to Grow 122

6.4 Push is Systemically Broken for Incident Notifications 123

6.5 Mission Stakeholder Empowerment 123
6.6 Anatomy of a Cyber Incident – Current Methods versus Proposed Architecture 127
6.7 Conclusion 129
VII – Conclusions and Recommendations 131
7.1. Conclusions 131
7.2. Recommendations for Future Research 133
7.3. Final Thoughts 134
7.4. Summary 134
Appendix A: Acronyms Used 136
Bibliography 138
Vita 143

List of Figures

Figure 1. Table 7.3, USI Action Matrix, from AFI 33-138 18
Figure 2. SCO Organizational Chart post-Communications Squadron Reorganization 21
Figure 3. Telephone Info Exchange Capabilities from *Power to the Edge* 31
Figure 4. Networked Collaborative Environment Capabilities from *Power to the Edge* 31
Figure 5. Flat-level or linear view of the proposed architecture. 43
Figure 6. Multi-base, multi-mission view of the proposed architecture. 45
Figure 7. A basic diagram of CLearn from Milcord [49] 48
Figure 8. SCOI Workcenter User Configuration Files 51
Figure 9. User configuration files and alerts for Mission A and Additional Duty B 52
Figure 10. A status request is made from the user agent to the local level server. 57
Figure 11. Local server responds with the status of local resources, inquiries top level server 57
Figure 12. Top level server responds, local level timestamps status and sends to user agent 58
Figure 13. Cyber asset ISCOM6YA fails. 59
Figure 14. Use Case Diagram Legend 82
Figure 15. Overall Use Case Diagram 83
Figure 16. Overall Class Diagram 87
Figure 17. Add New User Sequence Diagram 89
Figure 18. Add New User Communication Diagram 90
Figure 19. Add New User Activity Diagram 91
Figure 20. Create Status File Sequence Diagram 93
Figure 21. Create Status File Communication Diagram 94
Figure 22. Create Status File Activity Diagram 95
Figure 23. Mission Authority Delegation Sequence Diagram 97
Figure 24. Mission Authority Delegation Communication Diagram 98
Figure 25. Mission Authority Delegation Activity Diagram 98
Figure 26. Logic Flow – Adding a Dependency 102
Figure 27. Sequence Diagram for Adding a Mission or Asset Dependency 102
Figure 28. Communication Diagram for Adding a Mission or Asset Dependency 103
Figure 29. Activity Diagram for Adding a Mission or Asset Dependency 104
Figure 30. Flow diagram - Status Update 107
Figure 31. Flow Diagram - Updating to the TLNS 107
Figure 32. Sequence Diagram - Update Status File 108
Figure 33. Communication Diagram - Update Status File 109
Figure 34. Activity Diagram - Update Status File 110
Figure 35. Flow Diagram for UA startup and UCF load 113
Figure 36. Sequence Diagram - Monitor Status File Receives Alert 117
Figure 37. Communication Diagram - Monitor Status File Receives Alert 118
Figure 38. Activity Diagram - Monitor Status File Receives Alert 119

List of Tables

Table 1. AFSC Conversions (Partial List)..22

I. Introduction

1.1 Background

In a notional scenario presented by Sorrels [1], the confidentiality of a computer resource has been breached. Contained on this computer system was a database containing vital information regarding convoy operations in an active Area Of Responsibility (AOR). While this system compromise was detected, the process of trying to identify and then notify the users of the system was too slow and/or ineffective to prevent the loss of life and equipment when a convoy was later ambushed by attackers with inside information.

This situation, while hypothetical, expresses what could be considered a worst case scenario: our own information used against us leading to deaths of American service members. Insult is added to injury when personnel knew that this information was lost to the enemy but were unable to notify appropriate people of the compromise in order to prevent or minimize losses.

Our early 21st century world relies on technology and information. The pace of growth in this reliance on collections of 1's and 0's has exceeded system administrators' abilities to identify and notify downstream users. When there is a cyber incident, whether malicious or not, these system administrators have mere bread crumbs available to them to assist in locating the users of the data stored on their systems with a level of urgency commensurate with the impact to those downstream users' mission(s).

But more stark is the fact that most mission stakeholders, those who are responsible for the execution of our nation's business, do not understand exactly how the loss or compromise of their cyber-based resources impact their ability to perform their jobs. Where cyber-based assets fit into the mission capability of an organization requires a level of introspection that is difficult, time-consuming, and only truly valid until the next change in technology, responsibility, or leadership in that organization—a change that could easily occur before that introspective process is completed.

1.2 Problem statement

The primary concern when a cyber incident has occurred is to provide timely and relevant notifications to those missions that relied on the compromised system so that the stakeholders of those missions may take the proper actions to mitigate damage. To do this, a mechanism must be in place that executes this notification at a level of urgency commensurate with the mission impact of the compromise. To understand the mission impact to the loss or compromise of an information resource, mission stakeholders must understand how cyber assets fit into their overall mission dependencies.

The purpose of this research is to identify an improved method for communicating incident notifications in a more timely and relevant manner which includes both the missions directly impacted by the incident and the downstream consumers of those impacted missions.

1.3 Research goals

The research goals are:

- Identify and explain how the structure of installation level communications squadrons makes implementation of Air Force Instruction (AFI) 33-138, *Enterprise Network Operations Notification and Tracking*, unworkable for providing effective and timely notifications to enduring missions [2].

- Describe the concepts of "push" and "pull" communications and compare and contrast the two in their effectiveness and implementation (as applicable) in incident notifications.

- Demonstrate how pull notifications and/or publish and subscribe can eliminate many of the systemic problems inherent in AFI 33-138 [2].

1.4 Scope, Assumptions, and Limitations

A central goal in this research is to shift the focus of cyber incident notifications from system administrator-centric to mission stakeholder-centric. In keeping with much of the current literature regarding responsibility for flows of communication, the intent is to demonstrate theoretically that notifications should be a pull-based process rather than the current push-based process.

The scope of this research does not extend to a desktop-ready notification system. The concentration has been on exploring why the current processes do not achieve the goal of timely and relevant notification rather than real-world implementation. This extends both to communications theory and organizational implementation.

Further, the scope of this research does not extend to the topics of operations security (OPSEC) or potential security classification issues. It is acknowledged that a collection of operations information that a notification architecture would contain would, at a minimum, constitute the elements of essential information as defined in Joint Publication (JP) 1-02, *Department of Defense Dictionary of Military and Associated Terms* [3] and as such will need to be explored as an implementation detail.

This research does not take into consideration enduring missions that would include coalition partners from other nations. While there are a number of very good papers that touch on similar subjects, most notably [4], the level of difficulty in full-implementation of a model such as what is being proposed would jump exponentially.

The scope of relevance for this research is limited to messages reaching individuals with authority for missions who can use the message rather than those for whom the message does not pertain to. As an example, a message about System X would only be relevant in this way to those who use System X. Relevance of impact to the mission falls outside of the scope of this research.

This research is intended to provide the architecture for a decision support system—a single tool in a mission stakeholder's tool box to help that stakeholder make a decision. It does not pretend to take the place of human decision making, no more than a "Check Engine" light on a vehicle's dashboard takes control of that vehicle and drives it to the nearest mechanic.

The scope of this research does not explore the social and behavioral science implications of communicating an honest assessment of mission capability to interested parties.

1.5 Methodology

This is research is designed to be a proof of concept or feasibility study, with the focus centered on developing a theoretical notification architecture. Chapter IV will discuss the overriding considerations, the forms of modeling considered and either discarded or adopted.

1.6 Preview

This report is organized into seven chapters, with this first chapter being the introduction. Chapter II provides a literature review on cyber incidents, cyber incident mission impact assessment, missions, situational awareness, and mission capability. Chapter III steps through a series of investigative questions that begin from the research problem and then filter down into the component parts of that research problem. Chapter IV discusses methodology for this model and why UML was used. Chapter V draws out a variant of this system in UML 2 documentation, showing six basic use cases that capture the core of what the architecture is designed to do. Chapter VI makes a case for how, if implemented, this pull architecture is superior to the push scheme that exists in the framework of AFI 33-138. Chapter VII lists the conclusions that were reached and offers suggestions as to where additional research may be directed to further this work.

II. Background Information

2.1 CIMIA – Definition, Description, and Components

This chapter describes issues and literature critical to understanding the Cyber Incident Mission Impact Assessment (CIMIA) problem and the underlying reason why an architecture is needed that will allow for automatic targeted notifications of cyber and cyber-related incidents to enduring mission stakeholders.

CIMIA is a relatively new concept as a formalized process. Musman et al state, "We currently have very little capability to estimate the mission impact of cyber incidents." [5] Grimaila et al define CIMIA's purpose as "…to provide decision makers with timely notifications and relevant mission impact estimation, from the instant an information incident is declared, until the incident is fully remediated." [6] A key point here is that CIMIA is not autonomously making decisions. It is a notification process so that humans in charge of missions can make decisions. As is implied in this article, timely and relevant notifications of this nature are not being made at this time.

CIMIA also falls under the mission assurance umbrella of cyber operations. In Air Force Doctrine Document (AFDD) 3-12, *Cyberspace Operations*, it is clear that mission assurance remains one of the challenges to be conquered. The document goes on to say, "Mission assurance entails prioritizing mission essential functions (MEFs), mapping mission dependence on cyberspace, identifying vulnerabilities, and mitigating risk of known vulnerabilities." [7] While very well spelled out, absent both from this document and our normal day-to-day operations are the mechanisms to put the concept into practice.

Bargar notes that “Real-time risk management and situational awareness are essential to responding to a cyber crisis, as is the consideration of what national security missions are affected, potential cascade effects, and the prioritized approaches for restoration.” [8] Of great interest in the research for this notification system are the cascade effects which will be described in this research as what effect an incident has on downstream users. Bargar goes on to say that mission operations must be capable of continuing even if being attacked by natural and human adversaries [8]. As such, CIMIA could and should be a key component of real-time risk management.

CIMIA, broken into its high-level component parts, consists of three distinct elements.

- Cyber incidents, or things that happen in the cyber domain.
- Mission, or the performance of an activity that is communicated through commander’s intent, and
- Impact assessment, or what happened and what are its immediate and lingering effects.

For the duration of this report, mission and impact assessment will be combined and discussed as one as the two subjects are inexorably intertwined.

2.1.1 Cyber Incidents

The United States Air Force has attempted to provide situational awareness (SA) for cyber incidents within the context of AFI 33-138. The instruction segregates outages based on intent. Malicious outages are considered to be incidents. Non-malicious unplanned events causing down time or mission impact is an Unscheduled Service Interruption (USI). Planned events causing downtime or mission impact is an Authorized Service Interruption (ASI) [2].

From the perspective of a mission stakeholder, whether the cause was malicious or benign matters only in the level of urgency exhibited to the response. Regardless of intent, that stakeholder will need to respond. That response may be little more than a yawn should the issue be with a little used resource that is expected to be available in a matter of minutes. Or the response may resemble that of a theatre in which "Fire!" has been yelled if the resource was compromised with malicious intent and active data exfiltration is ongoing. In either instance, and in the plethora of scenarios that fall between those extremes, a mission stakeholder will need to evaluate what it means to their mission and take appropriate steps. For that reason this report will diverge from the AFI 33-138 definition and classify all situations in which one or more branches of the CIA triad (confidentiality, integrity, or availability) is broken, regardless of intent or pre-planning.

In "Computer Security: Art and Science," Bishop states that the CIA triad is the essence of what we are protecting with computer security. Bishop defines confidentiality as the "…concealment of information or resources." In short, confidentiality is keeping information out of the hands of those who do not need to know. Confidentiality includes knowledge of whether or not the information exists in the first place, which Bishop goes on to explain can often be as revealing as the information itself [9]. The lineage of the concept of confidentiality can be traced back to Bell and La Padula [10].

Bishop defines data integrity as "…the trustworthiness of data or resources…" Data integrity in a military environment can mean the difference between life and death. It is possible to breach data integrity without breaching data confidentiality. An example of this is if an intruder has gained access to a system and has write access but not read access, that intruder can write false data into the system without ever having seen valid data. Other examples of data integrity being compromised without a coinciding issue with confidentiality include malfunctioning software writing incorrect values into a data source and inadvertent false entries being made by a human operator [9]. The concept of integrity in this context goes back at least as far as Biba[1] [11].

Availability is defined by Bishop as, "…the ability to use the information or resource desired." The best example of attacks on availability are so-called Denial of Service or Distributed Denial of Service attacks (DOS/DDOS). In Distributed Denial of Service Attacks, Lau et al state that, "A denial of service attack is characterized by an explicit attempt by an attacker to prevent legitimate users of a service from using the desired resources." Availability breeches do not necessarily have to be external events. Inadvertent or purposeful removal of a single user or all users on a system will have the same effect as flooding the server with packets of data to the user or users who have been deleted. Losing availability can be through both malicious and inadvertent means and the impact to the mission is the same—legitimate users are blocked from being able to use a particular resource [9]. Malicious attacks on availability have been around as early as the

[1] It could be argued that Biba was discussing software code integrity rather than data integrity. In the Von Neumann computer architecture in which data and code are stored in the same memory space, inadvertent or malicious memory corruption may lead to either code and/or data integrity incidents.

so-called Morris worm in 1988 [12] [13], but the era of DDOS attacks can be traced to the well-publicized attacks of Yahoo!, America Online, and CNN in February 2000 [14].

2.1.2 Mission, Mission Capability, and Impact Assessment

"The mission" is one of the most often used phrases in the United States military. Everything is a mission. Recruiting a young man or woman and sending that person off to basic training is a mission. Feeding that service member, either in garrison or on the lines of combat, is a mission. Transporting that service member from base to base or one AOR to another is a mission. A plane dropping bombs on a target is a mission. Everything is termed as a mission.

JP 1-02 defines a mission as, "The task, together with the purpose, that clearly indicates the action to be taken and the reason therefore," and, "In common usage, especially when applied to lower military units, a duty assigned to an individual or unit; a task. [3]" It is that second definition that this research will concentrate on.

Keeping the mission going is one of the ways that success or failure is measured. The granularity of missions run from the smallest task to the overall goals and vision of the Department of Defense (DoD). Missions beget missions. Clusters of smaller missions make up larger missions. Most missions are dependent on multiple other missions. Very few missions are end points. For the purposes of this thesis, "mission" will be a generic construct. It could be as big as that of Headquarters, Air Force (HAF) or as small as an individual workcenter on a single base. The mission of dropping bombs on a target mentioned above is an example of the culmination of a long series of missions starting with recruitment and ending with the attempted (and likely successful) destruction of a valid target. The only requirement for it to be a mission is that it completes a task and somebody, that mission's stakeholder, is responsible for its execution.

All missions are not created equally. Some missions are continuous in nature. Others are performed just once and never performed again. While not specifically defined in JP 1-02, the term "enduring" is used repeatedly within the definitions in the publication. As an example, in the definition of a planning team it states, "The planning team is not enduring and dissolves upon completion of the assigned task [3]." For purposes of this thesis, the term "enduring mission" shall mean a mission that is performed on a regular basis by a specific organization and has manpower assigned to the accomplishment of that task.

Also for the purposes of this thesis, a generic enduring mission is a consumer of services, supplies, and personnel hours and provides an end product (physical items, another service, or both) to one or more other entities, most of which are other enduring missions.

JP 1-02 defines mission capability as, "Material condition of an aircraft indicating it can perform at least one and potentially all of its designated missions. Mission-capable is further defined as the sum of full mission-capable and partial mission-capable." The document further delineates when something is full mission-capable (FMC), partial mission-capable (PMC) or not mission-capable (NMC). As an example, full mission-capable is defined as "Material condition of any piece of military equipment, aircraft, or training device indicating that it can perform all of its missions…" In practice, FMC, PMC, and NMC are sometimes referred to respectively as Green, Yellow (or Amber), and Red [3]. Air Force Pamphlet (AFPAM) 63-128 further extends FMC, PMC, and NMC to all weapons systems [15]. In practice, FMC, PMC, and NMC can also be extended beyond physical items and into the abstract concept of mission, as demonstrated in Air Sovereignty and Air Defense missions [16] which will be discussed in more detail later.

The desired state for any mission is FMC. AFDD 1-1 states the third of the United States Air Force's three core values is "Excellence in all we do." As such, excellence demands nothing less than being fully capable of meeting mission requirements. For every mission, the desired state day in and day out is to be fully mission capable. Our *raison d'être* is to perform a mission. If not for the mission and the need to perform it, organizations and their Airmen would not exist. And because what a mission produces is consumed by another mission or missions, the mission capability of one organization spider webs into the missions of many other organizations [17].

Mission success or failure is ultimately the responsibility of a commander. A standard mantra in the military is exemplified by the words of Army Regulation 600-20, Paragraph 2-1.b., "Commanders are responsible for everything their command does or fails to do." [18] Commanders may delegate authority, but not final responsibility, for a mission or part of a mission through the communication of commander's intent. JP 1-02 defines commander's intent as, "A concise expression of the purpose of the operation and the desired end state…" [3]

Authority for missions may be delegated both hierarchically and non-hierarchically. Hierarchical delegation is similar to that of streams running into rivers and rivers running into oceans. Authority for the mission is subdivided along organizational lines as illustrated in AFI 38-101, *Manpower and Organization*. All of the squadrons that belong to the mission support group perform tasks that support the overall mission of a base, but those tasks are split out in logical ways (i.e. the authority for base security is delegated to the security forces squadron). At the squadron level it is broken up again with that authority further delegated. AFI 38-101 provides both a broader and more in depth discussion on standard Air Force hierarchy [19].

Missions can also be delegated non-hierarchically. An example is appointment to a wing exercise evaluation team (EET). EET members are subject matter experts from across the breadth of a wing or wing-equivalent that observe and document the performance of Airmen during the course of exercises. The wing commander tasks individuals in the various organizations under his or her command directly rather than making those individuals follow the chain of command up from where their normal positions are and up all of the echelons it would take to get back to the wing commander. When working in as a member of the EET, that individual works directly for (and under the authority of) the wing commander [20].

Impact assessment is a close cousin to battle damage assessment (BDA). JP 1-02 defines BDA as, “The estimate of damage resulting from the application of lethal or nonlethal military force. Battle damage assessment is composed of physical damage assessment, functional damage assessment, and target system assessment [3]." In short: How much damage did I do to my enemy and how much damage did his forces do to me?

Grimaila and Fortson note that damage assessment and mission impact assessment, while similar, differ in that a cyber incident requires an evaluation to be done based on how the damage may impede or could impede the ability to perform the mission. To get an understanding requires an overall look of all of the processes involved in mission success. This is far from a trivial task. It can be aided by using business process modeling, risk modeling, and any number of other tools. But this process is largely performed manually and is only as good as the depth of introspection applied to it and time available to perform it. And the dynamic nature of missions, both enduring and discrete, equates to a short useful shelf life for the work performed [21].

2.2 Communication Issues Relevant to Notifications

In *Improving Cyber Incident Notification in Military Operations*, the authors present five key limitations in the existing incident notification process [22].

- Inability to identify critically dependent downstream information resource consumers
- Failures in internal organization communication
- Lack of automation in the incident notification process
- The dynamic nature of organizations and their missions, and
- The determination of criticality is organization dependent

2.2.1 Identifying Critically Downstream Consumers

When an incident occurs, being able to contact the users of the system on which the incident has occurred would ensure that those users and the missions they are responsible for would know that they either did not have that resource (if an availability problem), had been possibly compromised by and shared with unauthorized recipients (if a confidentiality problem), or had been compromised and altered (if an integrity problem). This ability is not readily or reliably available in today's military networks. There are breadcrumbs scattered about, but there is no solid means to do this every time there is a problem.

Rather than being able to go directly from system owner/operator to system user, the Command, Control, Communications, and Computer (C4) NOTAM process in AFI 33-138 is utilized. There are solid reasons related to failures in internal organizational communication that will be detailed in the next subsection, but for now it is sufficient to say that the AFI 33-138 process does not perform as necessary to resolve the problem at hand [2].

Many have come up with different methodologies to try to overcome this problem, of which Camus [23] and Mission Service Automation Architecture (MSAA) [24] are two examples. Camus tries to bridge this gap by associating who uses a resource and where they work to the potential mission that is affected and a possible point of contact by utilizing (among other things) Lightweight Directory Access Protocol (LDAP) and system logs [23]. Stanley et al. proposed MSAA as a centralized database to track configuration of the network from which users with different levels of authority and responsibility of the network enterprise could share updates [24].

What is missing in this and nearly all of the other schemes developed is that it does not reach the granularity needed to get to the people responsible for the mission. As an example and as presented in the literature, Camus has the limitation of having no certainty that the message is going to reach the appropriate level of responsibility based both on a lack of granularity in the LDAP directories as well as not taking into account non-hierarchical missions [23].

What is also missing from both the schemes designed for automatic notification such as Camus [23] and MSAA [24] and the non-automatic notification scheme in AFI 33-138 [2] is the dynamic nature of today's modern military. There is no certainty that even if the right mission is picked and the right person is picked that the "right person" will be present to receive the message. The standard in-garrison work week for most Airmen hovers around 40 hours, yet many missions (and the drumbeat of attacks on our networks) carry on at all hours of the day and night on weekdays, weekends, and holidays. Issues that happen during the "normal" duty day may still not be picked up on a perfectly selective pushed message due to the intended recipient being gone due to illness, deployment, meeting attendance, meal break, or any number of other situations. The person nominally picked as the point of contact in case of incident may or may not be present when an incident occurs.

2.2.2. Failures in Internal Organization and Communication

Grimaila et al listed as a second CIMIA limitation the failure of internal organizational communication [22]. Undoubtedly internal organizational communication remains a problem as there is a broken communication linkage between those who receive the alerts and who need the alerts. Using the mandates of AFI 33-138, internal organizational communication is an issue that will be explored shortly [2]. Before arriving there, an examination of how we have allowed to keep the instruction in effect (of which compliance is mandatory) in the wake of massive organizational change must also be explored.

2.2.2.1 Communication Impairment Caused By Organizational Structure

AFI 38-101 lists the standard levels of Air Force organizations as HAF, major command (MAJCOM), numbered air force (NAF), wing, group, squadron, and flight. A set of flights makes up a squadron, a set of squadrons makes up a group, a set of groups makes up a wing, and so on. Per this AFI, it is the squadron that is the basic organizational unit in the Air Force [19].

As previously noted, incident notification in the United States Air Force is governed by AFI 33-138. As with all instructions, compliance with the instruction is mandatory. This means that all who are governed by AFI's (to include enlisted, officers, and government civilians) are required to comply with the instruction [2].

In AFI 33-138, different levels of the Air Force enterprise have different responsibilities that must be carried out. An example of this Table 7.3. from AFI 33-138. Who recognizes or is notified of an unscheduled service interruption (USI) has specific steps to perform in the notification process. Workgroup Managers (WM's) are at the bottom of this chain and the Air Force Network Operations and Security Center (AFNOSC) is at the top of the chain. Once the notification reaches the AFNOSC level and there is reason to notify down the chain, then a series of pushes occur in the form of a C4 NOTAM (in the form of e-mails or other messages) back down the chains. Pushes and pulls will be discussed in more detail section 2.5 [2].

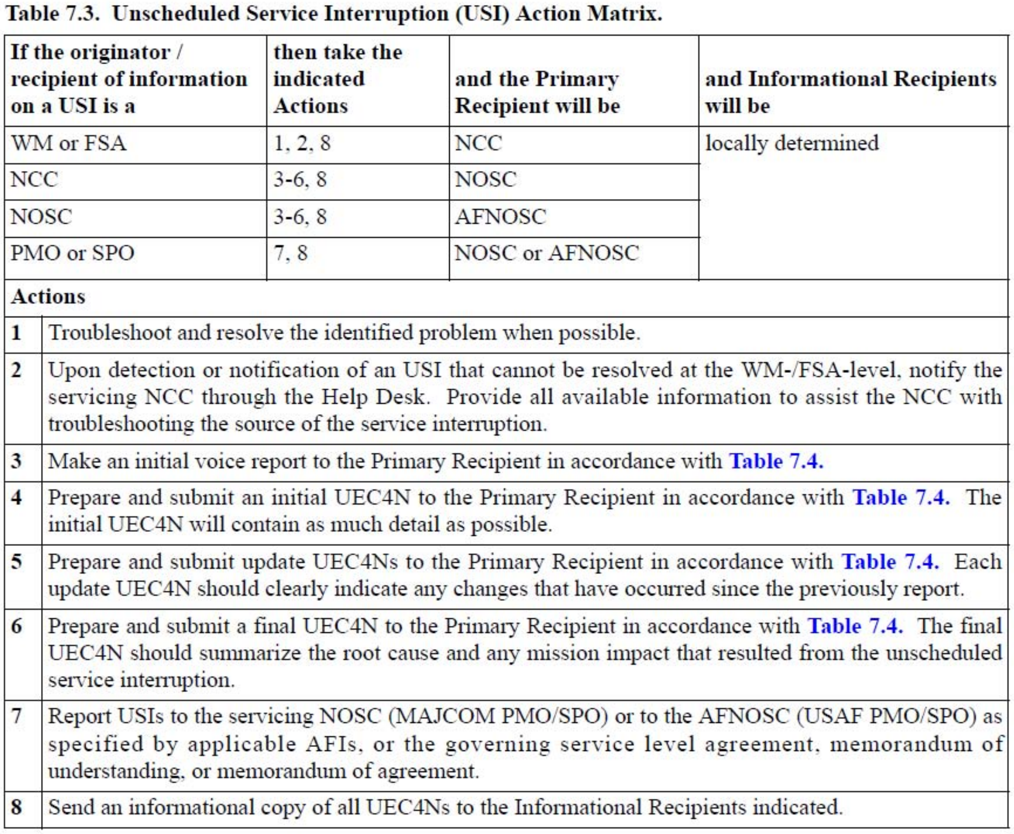


Table 7.3. Unscheduled Service Interruption (USI) Action Matrix.

If the originator / recipient of information on a USI is a	then take the indicated Actions	and the Primary Recipient will be	and Informational Recipients will be
WM or FSA	1, 2, 8	NCC	locally determined
NCC	3-6, 8	NOSC	
NOSC	3-6, 8	AFNOSC	
PMO or SPO	7, 8	NOSC or AFNOSC	

Actions	
1	Troubleshoot and resolve the identified problem when possible.
2	Upon detection or notification of an USI that cannot be resolved at the WM-/FSA-level, notify the servicing NCC through the Help Desk. Provide all available information to assist the NCC with troubleshooting the source of the service interruption.
3	Make an initial voice report to the Primary Recipient in accordance with **Table 7.4.**
4	Prepare and submit an initial UEC4N to the Primary Recipient in accordance with **Table 7.4.** The initial UEC4N will contain as much detail as possible.
5	Prepare and submit update UEC4Ns to the Primary Recipient in accordance with **Table 7.4.** Each update UEC4N should clearly indicate any changes that have occurred since the previously report.
6	Prepare and submit a final UEC4N to the Primary Recipient in accordance with **Table 7.4.** The final UEC4N should summarize the root cause and any mission impact that resulted from the unscheduled service interruption.
7	Report USIs to the servicing NOSC (MAJCOM PMO/SPO) or to the AFNOSC (USAF PMO/SPO) as specified by applicable AFIs, or the governing service level agreement, memorandum of understanding, or memorandum of agreement.
8	Send an informational copy of all UEC4Ns to the Informational Recipients indicated.

Figure 1. Table 7.3 (USI Action Matrix) from AFI 33-138 [2]

The AFI, as written and still in effect as of the writing of this document, has not caught up with the changes made in the Air Force since the instruction was written.

There was a definite echelon that existed within Air Force wings that flowed from the network providers (typically communications squadrons) and their network control centers (NCC's) to the organizations that consumed the resources. Within each consuming organization there were a number of individuals identified as WM's. By instruction, WM's belonged to the 3A0x1 Air Force Specialty Code (AFSC), also known as Information Management—an AFSC formerly more associated with administrative tasks and/or civilian secretaries. As WM's, they served the role as the initial point of contact for computer issues for users in their organization prior to initiating a request through the base help desk—the Air Force equivalent of first tier technical support. In theory, these individuals knew best what it was the computer users in their organizations were using in terms of internal and external systems [25]. In terms of sheer numbers, the vast majority of WM's would be assigned to the squadron level—again, as defined in AFI 38-101, at the basic level of Air Force organization [19].

On December 28, 2005, the Department of the Air Force released "Programmed Budget Decision 720: Air Force Transformation Flight Plan," or what was known in common usage as PBD 720 [26]. PBD 720 was a "draconian plan" (as described by Donnithorne) hatched by the Secretary of the Air Force and the Air Force Chief of Staff to eliminate 57,000 military and civilian personnel positions, 40,000 of which were Airmen. The goal of PBD 720 was to free up money to repair, upgrade, and/or replace portions of the service's legacy inventory of aircraft [27]. Of these 40,000 Airmen, over 8,000 were in Communications and Information AFSC's, that is AFSC's that at the time belonged to the 2EXXX, 3AXXX, and 3CXXX AFSC families. This forced a restructuring of the Air Force communications enterprise that removed authorizations from wing-level communications squadrons and relocated them to MAJCOM and higher level organizations [25].

This started a ripple effect which prompted if not facilitated a major change in the structure of base-level communications squadrons, as illustrated in Figure 2. This change made it so NCC's as monolithic entities no longer existed. Communications squadrons restructured from a four flight structure to a two flight structure in October 2008. Prior to that, NCC's fell under the operations (SCB) flight and consisted of the help desk, a set of server room back shops, an information protection shop, and a technical control shop. All belonged to the basic umbrella of SCBB, subordinate to the SCB Flight Commander. The NCC section chief, typically an enlisted person in the rank of E-7 or above (also known as a Senior Non-commissioned Officer or SNCO), would report to the SCB flight commander. That flight commander in turn would report to the Communications Squadron commander [25].

When the squadrons went from a four flight to a two flight structure, major changes were made to the structure of the NCC to the point of the NCC disappearing as a separate entity. The new operations flight (SCO) absorbed most of the former SCB's responsibilities as well as most of the maintenance flight and a few other minor pieces. The help desk was rebranded as the Communications Focal Point (CFP) and was assigned to operations services section (SCOS). At the same time, WM's (now known as client system administrators or CSA's) were removed from non-communications squadron units and were gathered to form the nucleus of the new CFP. Most of information protection and the entire server room back shop mission were incorporated into a new operations section of the operations flight (SCOO). Technical control personnel, those responsible for circuits coming into the base and the routers, switches, and encryption devices forming the network's backbone were combined with telephone maintenance and formed the new communications infrastructure section (SCOI). The position of NCC Section Chief was done away with and all of the leaders of these new sections then reported directly to the SCO Flight Commander [25].

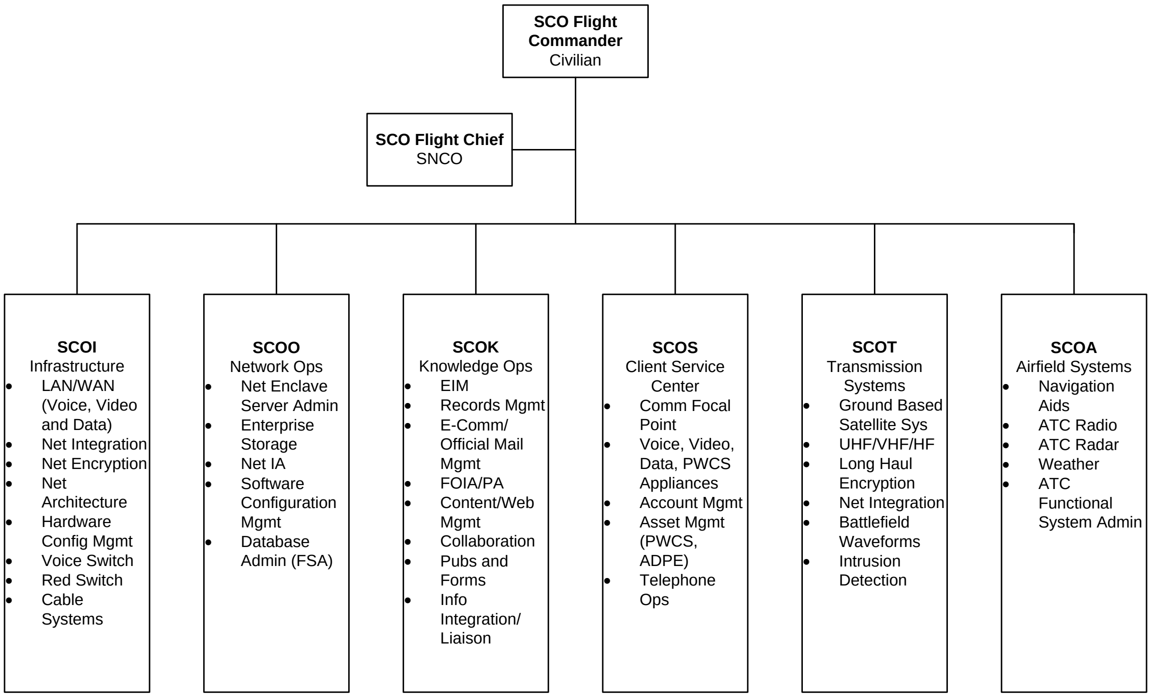
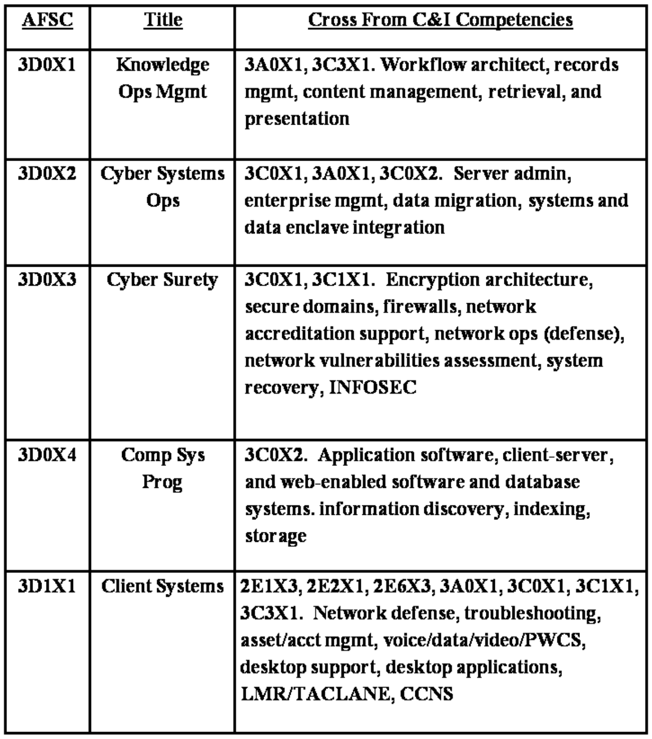


Figure 2. SCO Organizational Chart post-Communications Squadron Reorganization [25]

If all of this were not confusing enough, between October 2009 and January 2010 the Air Force also reorganized the AFSC's of the personnel who work in what had been the NCC. Gone were the scattering of 2EXXX (Electronics Maintenance), 3A0X1 (Information Managers) and 3CXXX (Communications and Computer Systems) personnel who worked in the NCC and instead all were rebranded and repurposed as 3D0XX (Cyber Operations) and 3D1XX (Cyber Systems) personnel, with both 3D0XX and 3D1XX AFSC families having additional sub-specialties. This AFSC reorganization not only changed names and shuffled people, it also fractured old AFSC's and divided the billet authorizations to be spread to up to 5 of the new rebranded and repurposed AFSC's. This meant that some of the experience that had been in the former NCC was removed to other jobs outside of the complex and personnel that were previously on the outside looking in found themselves in what was the former NCC [28].

Table 1. AFSC Conversions (Partial) [28]



AFSC	Title	Cross From C&I Competencies
3D0X1	Knowledge Ops Mgmt	3A0X1, 3C3X1. Workflow architect, records mgmt, content management, retrieval, and presentation
3D0X2	Cyber Systems Ops	3C0X1, 3A0X1, 3C0X2. Server admin, enterprise mgmt, data migration, systems and data enclave integration
3D0X3	Cyber Surety	3C0X1, 3C1X1. Encryption architecture, secure domains, firewalls, network accreditation support, network ops (defense), network vulnerabilities assessment, system recovery, INFOSEC
3D0X4	Comp Sys Prog	3C0X2. Application software, client-server, and web-enabled software and database systems. information discovery, indexing, storage
3D1X1	Client Systems	2E1X3, 2E2X1, 2E6X3, 3A0X1, 3C0X1, 3C1X1, 3C3X1. Network defense, troubleshooting, asset/acct mgmt, voice/data/video/PWCS, desktop support, desktop applications, LMR/TACLANE, CCNS

In AFI 33-138, NCC and WM are two steps in the process for getting C4 notifications to the user populace. But in practical terms, neither the NCC as an entity nor WM as position exist as intended in the instruction. Some may extend the concept that the SCO Flight Commander now is and acts for the NCC. This is not a reasonable conclusion when you look at the other responsibilities that individual has, many of which extend to issues such as safety of flight [25]. And when the WM's were removed from the squadrons, there was no plan that was put in place that identified who would replace the WM's role at the basic level of Air Force organization [28].

Yet in accordance with AFI 33-138, compliance with the instruction is mandatory. The Air Force did not keep the established and/or documented notification scheme up to date while undergoing the significant structural changes through changes or updates to the governing instruction. In itself this is a form of failure in internal communication—the rules of road do not reflect the road surface.

Beyond the fact that compliance has not been achieved with the published instruction, there are two important things here which complicate and frustrate internal organizational communication. First is that there are no longer individuals specifically trained to handle pre-first tier support in the basic building block of the Air Force's organizational echelons. And second, an unfortunately piece of collateral damage of the AFSC restructuring is that experience levels have been inadvertently reduced inside the structure of what once was the NCC.

If we temporarily set aside the fact that today's communications structure does not match that which existed when AFI 33-138 was written [2] nor does it match the structure dictated in AFI 38-101 [19], the fact that there is no designated person in the basic building block of Air Force organizational structure to receive messages from the communications squadron demonstrates the potential for a communications breakdown.

As part of the investigative questions this will be explored in additional depth where, even if there had not been this tsunami of organizational change within the communications hierarchy that there are major failures with internal organizational communication.

2.2.2.2 Communication Impairment from Mission Execution Ignorance

As part of the C4 NOTAM process as outlined in AFI 33-138, an entry must be made in the C4 NOTAM to describe the mission impact of the incident. As mentioned above, removing WM's from the squadrons eliminated what was somewhat of a liaison between the communications and operations. This issue renews itself as a failure in organizational communication when it comes time to determine mission impact.

When the notification leaves the network operations community to any of the other user communities (operations or logistics as an example), problems start to occur. As noted by Hale et al., there is a chasm that exists between the network operations communities and the communities they support. The network operations community is responsible for all aspects of C4 systems but operations, as an example, sees C4 capabilities as a utility [29]. In accordance with AFI 33-138, it is the network operations personnel who are tasked with collecting mission impact data from the user communities for assessment at higher headquarters when an incident occurs [2]. When it reaches higher headquarters, a network operator at that echelon briefs the commander in charge, usually with representatives of the other user communities present for analysis and opinion.

Tinnel et al. speak to this kind of functional gap and asymmetric dependency between operations and information technology personnel. The same personnel who are trying to defend an active attack or a catastrophic hardware or software failure are simultaneously attempting to assess mission impact. And rather than just collecting the data, often it is network personnel who are also trying to measure that damage because the users of the affected resource(s) do not understand how important the resource(s) are to their ability to accomplish the mission [30]. As is the case when signal transfers from one media to another, it is possible that distortion or signal loss may occur in the form of lost or misinterpreted mission impact assessment as it goes from the operations communities to the communication community and back again.

2.2.3 Lack of an Automatic Notification System

As Alberts & Hayes explored in *Power to the Edge*, we no longer have the luxury of waiting for information [31]. This includes being notified automatically when a cyber asset has an incident. There is no such system available today. AFI 33-138 dictates the use of what is a very manual and human-intensive process.

Creating an automatic notification system is far from a trivial task as much of what has been discussed to this point has to be accounted for. Emphasis of such a system needs to be placed on downstream users, but not just those directly using the system which was involved in the incident, but those missions who are the consumers to that which is produced by the mission that was subject to the cyber incident. Missions do not work in a vacuum. Connecting the spider web of missions to other missions will be a daunting task.

Attempts have been made to create automatic notification systems. The most prevalent of these in this research was Command and Control Remote Monitoring System, or C2RMS. C2RMS, a replacement for Master Control Panel, was designed to aid in monitoring availability of cyber assets within the Combined Air Operations Center, or CAOC. Additional features were added to help extend its usefulness outside of the CAOC. While useful in that environment, it did not include monitoring of confidentiality or integrity, nor did it scale well to account for the web of downstream missions [32].

Grimaila et al point to a pull-based automatic incident notification as being a way to improve the cyber incident notification process. They suggest that a "central authority" be put into place that, while not spelled out directly, subscribes to information sources that they use and within the message received there is notice if there is an incident. By doing periodic pulls, checking for status goes on continually and, again without directly spelling it out, the periodic checks fall inside the typical noise of the network. Pull is not as loud as pushed messages when an incident has occurred, denying the adversary the opportunity to do their own BDA on a potential attack [22]. This falling within the noise of the network and scheme for automatic pulling became one of the cornerstones of the eventual direction for this research.

2.2.4 The Dynamic Nature of Organizations and Their Missions

Grimaila et al said it best when they said, "Organizations are inherently dynamic entities. As the organization, the organizational mission, or organizational mission processes change; the information resources requires to support its mission are also likely to change [22]." Alberts (and his various co-authors) acknowledge this in their writings and the need for organizations to live out on the edge rather than in the current hierarchical format that exists today. Cyber has only increased both the speed and the level of complexity that those with authority and/or responsibility for missions face [31] [33] [34].

2.2.5 The Determination of Criticality is Organization Dependent

Grimaila et al discuss the connection of criticality of a particular system to the successful completion of a mission varies based on the organization [22]. This can be extended with the Alberts & Hayes discussion of decentralized execution [31]. Beyond criticality being organizational dependent, it is also situational dependent based on what other tools are available to the person with authority and/or responsibility for the mission.

2.3 Situational Awareness (SA)

CIMIA is deeply rooted in SA. In the Endsley model, SA is defined as, "…the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future. [35]" In CIMIA, a notification of a cyber incident helps develop the SA of the mission stakeholder(s), providing them with a perception that an incident has occurred. It is then with a full accounting of the status of the other elements of their mission that comprehension of how this will affect their mission can take place. This cannot be underestimated—cyber for most missions will be a sliver of the big picture. While we are undoubtedly more dependent on information systems now, they are still only a tool towards overall mission accomplishment. Even in the most technologically based missions and objectives, people remain at least on par with the technology in terms of mission accomplishment.

The importance of SA for a mission stakeholder cannot be minimized. As Endsley points out, SA is an important element in aircraft, military, and civilian pursuits. Incorrect or incomplete SA can lead to massive loss of life in a worst case scenario, as demonstrated with the shooting down of an Iranian civilian airliner by the USS Vincennes [35].

As this research is aimed towards the development of an architecture that will provide automatic notifications of cyber incidents and their related downstream consequences, the concept of SA is an enormous aspect of the continuing project.

Endsley's model has three levels of SA. Level 1 is when awareness of an event is achieved. Level 2 is taking action based on what is perceived. Level 3 is being able to project future actions based on what is known. Throughout the remainder of this paper Endsley's model of SA (and the respective levels) will be referred to as EMSA Levels 1 to 3 [35].

The EMSA Levels are analogous to the difference between a beginning chess player, an experienced amateur, and a grand master. The beginner sees what his opponent is doing, but is oblivious to making proper moves in response. An experienced amateur knows both what is going on and what is a reasonable response to his or her opponent. The grand master not only knows the proper current play, but can project multiple turns in advance what the consequences of a particular move may be.

The purpose of the architecture that will be proposed is to achieve Level 1 EMSA for those with authority or responsibility for a mission, known for the remainder of this document as mission stakeholders. Leaders inherently work at EMSA Levels 2 and 3 [21], but tackling a system to provide those levels of EMSA falls well beyond the scope of this research.

2.4 Workflow and Business Process Modeling

Grimaila and Fortson discuss process modeling as a formal way to determine mission impact assessment [21]. A significant number of process modeling schemes and languages in the relevant literature were studied, evaluated, and ultimately moved on from a number of process modeling schemes through the course of this research.

Workflow and business process modeling comes in many flavors; each molded and formed to best meet a specific set of outcomes for a specific set of users. Among these are Department of Defense Architecture Framework (DoDAF) [36], Petri nets [37], Yet Another Workflow Language (YAWL) [37] [38], Integrated Definition (IDEF) [37] [39] [40], and Unified Modeling Language (UML) [41] [42].

Some modeling schemes work better for linear processes, others are more suited for operations that loop. Models such as DoDAF, IDEF, and UML have multiple ways to different aspects of the workflow process based on what is important to the viewer and the part of the project or plan that is of most interest.

UML version 2.0 was ultimately used as a starting point for this research. UML is an object-oriented approach to process modeling, and is one of the standards for formalized systems engineering [41]. It makes available a series of different types of drawings that are suited to getting into the fine details of a project, allowing for programmers and users to better connect in the design process, eliminating much of the stumbling blocks associated with designing and implementing a new system. The rationale for choosing UML was chosen for this research will be discussed in more depth in Chapter IV.

2.5 Pushing versus Pulling

Alberts & Hayes spend a good deal of time in *Power to the Edge* discussing the vast differences between pushing and pulling information. Pushing information is similar to calling somebody on the phone. You have information that somebody else needs, but the only way you can communicate that information is if you know their phone number and they are on the other end of the phone when you call. Figure 3 graphically illustrates those shortcomings. In contrast, post and smart pulling in a networked collaborative environment, as shown in Figure 4 allows for the publishing of that information to a centralized location and those who are interested in the information can pick it up at a time and method of their choosing [31].

The method of incident notification prescribed in AFI 33-138 is effectively a long series of pushes. Pushes occur from originator to AFNOSC and then as appropriate from the AFNOSC back down to individual users of the system [2]. Harkening back to the push as described as a telephone exchange, you have to have the right number and the person needs to be present to effectively communicate in a push. Further discussion on push versus pull when notifications arrive at the base level and need to get to the user level will occur in Chapter III and again in Chapter VI.

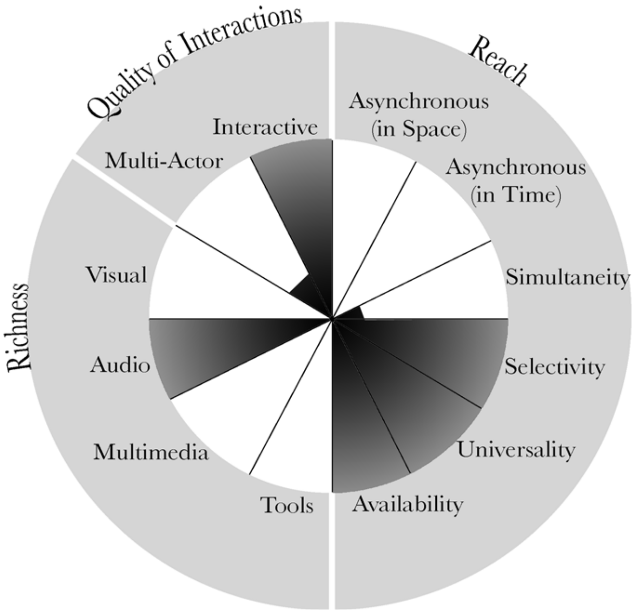


Figure 3. Telephone Info Exchange Capabilities from *Power to the Edge* [34]

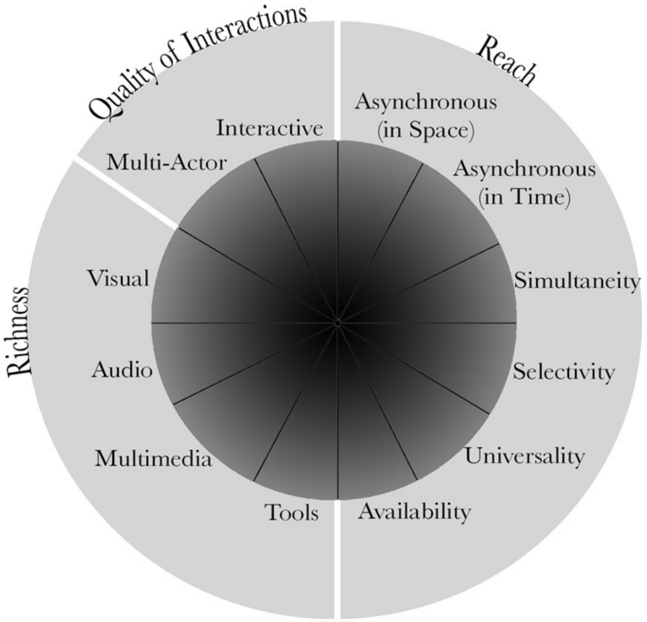


Figure 4. Networked Collaborative Environment Capabilities as shown in *Power to the Edge* [34]

The commercial Internet has embraced pull as a viable technology. Really simple syndication (RSS) utilizes publish and pull. A publishes posts an extensible markup language (XML) file on the Internet with a list of links to additional files that they have published. A user then views the XML file and chooses what it is they are interested in [43]. Facebook and Twitter work in a similar fashion in that, by gaining friends or followers, you are subscribing to their feeds. If you are online, you periodically send a request to either service that tells them to send anything new from any user you have subscribed to.

III. Investigative Questions

The purpose of this chapter is to look with additional depth at the overall research problem, apply what was learned from the related literature, and provide the foundation for an architecture that is capable of meeting the goals of the research.

The primary problem questions whether timely and relevant notifications can be received by mission stakeholders who need to receive them. This is the crux of achievement of Level 1 EMSA in a CIMIA environment.

Extending from this primary research question are three distinct but interrelated branches of investigative questions. The first branch explores why the current method of notification does not work, the theory suggesting change, and an explanation how a replacement architecture might function. The second branch follows on from this replacement architecture and discusses at a reasonably high level how notifications made in this scheme would be more effective in achieving Level 1 EMSA. The third branch briefly discusses some of the high-level details that are missing and acts as a preview for Chapter VII.

The research question and investigative questions are the foundation on which the methodology, use case, and case for action discussions in Chapters IV, V, and VI, respectively.

3.1 Research Problem: Can timely and relevant incident notifications be made to enduring mission stakeholders?

The various literature sources cited in Chapter II vividly reflect that we are not consistently making incident notifications in a manner that would be considered timely and/or relevant. This problem deeply rooted in the five limitations noted in Grimaila et al. [22] as enumerated also in Chapter II. The problems are made more severe by the aftermath of PBD 720, the subsequent reorganization of Air Force communication squadrons, and the removal of WM's from operational squadrons. Enduring missions appear to have the same problems as non-enduring missions in terms of receiving these notifications in a timely and/or relevant manner.

3.2 Primary Investigative Question 1: In accordance with AFI 33-138, C4 NOTAMs [2] are the means in which incident notifications are transmitted [2]. What is keeping C4 NOTAMs from consistently being timely and relevant?

There are many things reflected by the literature [21] [22] [24] [29] [30] [44] that illustrates why C4 NOTAMs do not achieve timely and/or relevant notifications and that this is not a new development. Information system owners do not have a means to track all of the missions that use that system as a mission dependency. These information system owners also do not known with any level of clarity exactly how important their system is in mission accomplishment. As such, there is no readily available means to make timely and relevant notifications directly to users.

With no automatic means of making these notifications, we are left with the generalized notification process as detailed in AFI 33-138 to communicate breakdowns in information and system confidentiality, integrity, and availability. Of the four types of C4 NOTAMs detailed in the instruction, Informative, Unscheduled Event, and Scheduled Event types that could be associated with CIMIA notifications. The fourth, Summary, is

[2] The term NOTAM is used repeatedly in both military and civil aviation. NOTAMs are issued as advisories, guidance, and warnings. C4 NOTAMs are specifically centered on Air Force network operations.

designed as a reply rather from lower echelons to higher echelons rather than a notification that there is a problem [2].

This report discussed the restructuring of both what was formerly known as the NCC and the AFSCs of the individuals who worked in that organization during Chapter II [25] [28]. Assuming though that the structure as intended still existed, additional issues still abound with the means in which notifications theoretically were carried out. C4 NOTAMs were and are designed to be transmitted via e-mail or eTANG at a classification level commensurate with the information contained within them. This transmission has the effect of a bidirectional information push between higher echelons to lower echelons, with direction starting up from the discovering organization to the top echelon and then broadcast down to the lower echelons for operational user notification [2].

Also assuming for a moment that all notifications would be done at an Unclassified clearance level, an individual at the NCC level would receive the notification and be responsible for sending out the notification to the WM level via e-mail--another push [2] into the noisy e-mail environment [12]. An enlisted technician, typically a subordinate of the server room or information protection shop non-commissioned officer in charge (NCOIC) and likely of the rank of E-5 or below has to make a decision as to which WM's should get the notification, if any.

The rank and hierarchical position of this technician is significant. With 10E5 promotion cycle (that is, those selected for promotion to the rank of E-5 during Fiscal Year 2010), the "average" enlisted Airman selected for promotion to the rank of E-5 during this promotion cycle will have approximately 4.55 years of time in service. Put more simply, that individual has been in the active duty military for that length of time [45]. That 4.55 year total includes time spent in basic training, initial technical skills training, professional military education required for promotion to E-5 [46], skills training required to meet the commercial certification requirements of DOD 8570.01M [47], approximately 135 days of vacation time, and any number of other tasks in line with being in the military—all things that can and usually do remove that Airman from their primary duty. This time away from the work environment hampers the ability for the Airmen in these duties from gaining the requisite experience and corporate knowledge to be able to associate the impacted system with the most likely organizations who may use that system. Yet in an average worst case scenario, this is the expectation.[3]

The technician receiving the incident notification then has a decision to make. If the technician decides to send the message to all of the WM's and it is a system that is used by only a handful of missions, then the technician sending the message has introduced an additional level of noise into the system. Libicki's discussion of noise as an information warfare tool noted that in a noisy environment we run the risk of eventually filtering out not only the noise but the signal that is sent in that noisy environment. Or more plainly, after a while WM's likely will just filter out all of the messages because they hardly ever pertain to the missions they support [12].

[3] An argument could be made advocating putting an individual with more rank and/or experience into the position that receives the C4 NOTAMs. However, the answer to this argument is that in the wake of the PBD 720 reductions, the next person in this young E-5's chain of command is likely either the NCOIC of the workcenter or the Section Chief [46]. Each of these positions, as dictated by applicable instruction, are tasked with the business of managing personnel and building teams and either position would allow the time necessary to gain and maintain the requisite level of insight to effectively filter notifications.

If instead that technician attempts to narrow the scope of the user base to whom the notification is sent, that Airman runs the risk of missing organizations that need to receive this notification. While the WM's receiving the notifications are more likely to respond because they receive less noise than in a full broadcast situation, the consequences of the notification not being received can be significant [12]. Expecting this average, newly-minted E-5 to be able to identify every enduring mission on their assigned installation, every system each of the missions use, and the level of criticality to the using missions approaches ludicrous.

A similar problem exists when the notification reaches the WM. Assuming for a moment that they receive the information and do not filter out the notification, they too have the same type of decision to make. It is anticipated that WMs understand their squadron's mission with a finer level of granularity than the technician at the NCC. However, WM tasks were but a fraction of the overall job responsibilities of most personnel carrying the WM designation as defined in the 3A0x1 Career Field Education and Training Plan [48]. And while in the NCC there was a rank expectation of the individual receiving the notification was in the realm of E-5, a WM's rank was dictated by the squadron's manning documents and the person receiving the notification could have held a rank in the lowest echelon of the enlisted corps. So again the choice becomes send it to all users, use *a priori* knowledge to vector the notification only to select people, or decide that the squadron wasn't involved in that line of work and discard the notification all together. In the hands of the most junior of Airmen, there is no telling what would be the final outcome of that message.

Also missing, as mentioned in Chapter II, is how to account for the dynamic nature of military service. In the best case where the E-5 receiving the notification correctly identifies all units that use the system affected by the incident and gets the notification to the WM's for those units, it is then up to the WM to find the individuals within the unit that use the system. Even a regularly updated list of systems and their users will fall prey to individuals gone for the reasons enumerated back in Chapter II. Subtract WM's from the units as we have done in the aftermath of PBD 720 and now the focus is shifted directly to a young E-5 doing their best with a nearly impossible task.

Systemically, push as implemented here is a losing proposition. Noise theory as extended to cyber by Libicki [12] and implemented on top of the structure established in AFI 33-138 [2] creates an unachievable scenario even if the structure of communications hierarchy existed as it did when the instruction was first published. With the change to the structure of communications squadrons and the pulling back of the WM's into the communications squadron, the problem of noise and lack of *a priori* knowledge is amplified. That technician at what would have formerly been known as the NCC level now is in a position of either broadcasting to all base users about the problem (increased noise) or has to have even finer *a priori* knowledge as to who might be using the system that has been subject to an incident which compromised its confidentiality, integrity, or availability.

Missing from this scheme, and something previously mentioned in Chapter II as a missing feature of Camus, is the ability to zero in on the right individual who needs to be notified in the case of an incident. Non-hierarchical missions create havoc with such a system.

An example of the potential chaos created by this can be illustrated in a fictional example. Two individuals in a squadron are assigned to a non-hierarchical mission. The individuals work in separate flights within the squadron (for purposes of this example they work in Alpha Flight and Bravo Flight). The one assigned to Alpha Flight has recently interacted with the system but is sent on a very short notice temporary duty assignment out of the country. An incident occurs and Camus lines up the individual who accessed the system with belonging to the Alpha Flight. The system owner attempts to notify the individual in Alpha Flight, but the e-mails go unanswered and phone calls to Alpha Flight are met with a level of confusion and ignorance expected when co-workers are not involved in the mission. Meanwhile, the individual in Bravo flight is oblivious to the problem because Camus had not identified that individual as a usable point of contact. Any solution to the notification problem must include the ability for all mission stakeholders to have the ability to achieve Level 1 EMSA when the authority for the mission falls to multiple personnel, whether working together or separate parts of the unit or base.

Also missing from this scheme is how to notify downstream users. Provided a timely notification is received by a mission dependent on a compromised cyber resource, there is no mechanism in place to alert the next layer of missions. This is what Bargar referred to as cascading effects [8]. As discussed at length by Hale et al. [29], cyber resources A, B, and C go into mission D. The product of mission D goes into mission E.

If one or more of the three cyber resources were to be compromised, it is possible that the current or previous output of mission D will also be affected. If mission D were responsible for another cyber resource, and the previous incident affected the CIA of the cyber resource mission D provides, then the full AFI 33-138 push notification process starts from the beginning. Otherwise, there is no formal mechanism available besides *a priori* knowledge of all of the missions dependent on mission D to get the word out of the potential issue.

Rather than push, current communications theory as championed by Alberts and Hayes would indicate that this would be a good place to implement pull [31]. If the owner of the data system or the appropriate Air Force level organization knows that there is a break to the CIA triad, then either of those parties could publish the fact that there is a problem and those who need to know that information could use an agent to pull down that status. As no enduring mission acts in a vacuum, that mission could then publish its status and/or a warning given the previous break to the CIA triad. Their downstream users would also use the agent to pull that status and be empowered to take appropriate actions as well.

3.2.1 Investigative Question: How would pull improve the notification process?

Pulling notifications, or publish and subscribe, would eliminate numerous middle layers of communication, providing a more direct route from data system or mission owner to dependent enduring mission. In the current AFI 33-138 construct, multiple humans are involved as the message travels from echelon to echelon. Each step where a human is involved slows the notification process. At each step there is the potential for the introduction of noise and/or imprecise filtering of messages, reducing the likelihood that the message will reach the intended recipients.

When moving to a pull scheme, the middle layers disappear. The only parties involved in the process are the mission and/or data resource reporting the status change and the missions dependent on that provider. If implemented properly, an architecture would exist to allow a place for those status messages to be published to and a means to retrieve the status message in a time frame consistent with the importance of that status update.

Through the elimination of multiple human layers and implementation of what effectively becomes a subscription to the publishing of statuses, timeliness and relevance increases provided that those who have been delegated the authority for these missions can identify what resources their mission uses (both cyber-based and/or the product of other missions). In this regard the onus moves from the communications personnel (or A6 as listed in AFI 38-101 [19]) to those performing the enduring missions and/or providing management of the individual cyber resources.

3.2.2 Investigative Question: What would such an architecture look like to support the "publish and subscribe" function?

A multi-level architecture should be used, with a minimum of three levels.

The architecture starts at a centralized top level server. This top level server stores primarily the status of cyber assets that are used at more than one base or base-equivalent. It is located within an organization that has an adequate amount of oversight of current cyber threats and attacks and personnel at that organization may manually update the status if there is a known or suspected cyber incident occurring with that particular cyber resource. The top level server receives updates from local level servers when status changes occur.

Local level servers act as the middle man between user agents employed by mission stakeholders and the top level server. Local level servers reside with the local base's communications infrastructure. The local level server is responsible for:

- Storing local level cyber asset status
- Storing local level enduring mission status
- Storing local level enduring mission configuration files
- Performing periodic pings/data inquiries to local level cyber assets to ensure availability (similar to Command and Control Remote Monitoring System (C2RMS) [32])
- Performing manual status updates of local cyber assets when warranted
- Requesting and storing non-local cyber asset status from the top level server
- Publishing the status of local level cyber asset statuses to the top level server

The local level server requests a new status from the top level server when it receives a request from a user and that user needs a more recent status check than the base level server has on record. When the new record comes back from the top level server, the local level server then communicates the most recently retrieved status.

The user-level agent acts as a status checker for the user running the agent. This agent may monitor and update multiple enduring missions depending on the responsibilities of the user running the agent. For each enduring mission that user is responsible for, statuses are retrieved from the local level server for that enduring mission's dependent missions. Those statuses are retrieved in a time interval commensurate with the importance of that dependent mission on the execution of the monitored enduring mission as identified by the dependent mission leaders based on commander's intent. When there is a change to the status of a dependent mission, the agent alerts the user. In the initial implementation, it is up to the mission stakeholder to decide whether or not to update the status of the enduring mission affected by the change in dependent mission status based on the mission stakeholder's SA.

Figure 5 shows a linear view of the proposed architecture. Arrows show data flow from the user agent up to the top level server and back. Items inside the logical blocks are specialized functions that the various levels perform.

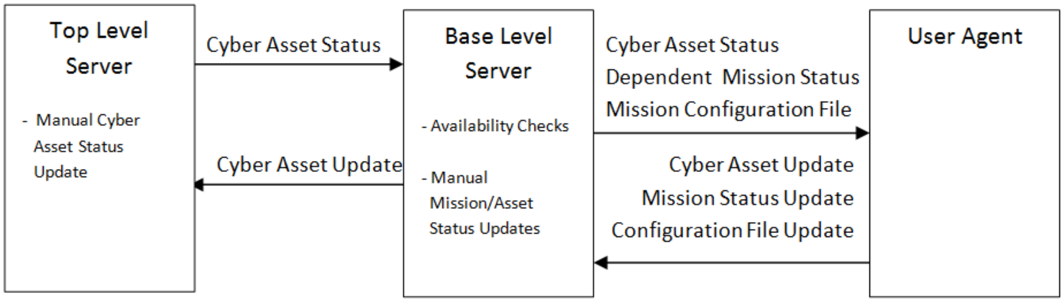


Figure 5. Flat-level or linear view of the proposed architecture.

Figure 6 represents a multi-base, multi-mission view of the same system. Local level servers, as shown in this figure, and the optional intermediate level servers (discussed later) are the key for system scalability. Local level servers manage all of the traffic for the symbiotic relationships that occur within the confines of an Air Force wing.

The local level servers publish the status of local cyber assets to the top level server and retrieve and cache the status of cyber assets from other bases via the top level server. The data is cached and kept on a timeframe commensurate with the mission on the local base that needs the most current information of a particular cyber asset. Even in cases where multiple users on a base all need to know the status of a single cyber asset, only a single request goes to the top level server on a cycle equal to that of the mission that needs that most current information. This reduces the load on the top level server and associated network architecture with eye towards scalability, yet still meets the mission requirements of all of the dependent missions on a particular base.

Scalability could also aided by maintaining a limit on the size of data being transmitted. Actual size is an implementation detail that can be developed further by future research, but should be based on the fields present in the framework discussed in Chapter V. The ideal situation is that these data transfers are so small and so unobtrusive that eventually all of this traffic will blend into the much larger streams of data flowing through our networks.

Intermediate levels could be put between the local and top level servers in places where geography and installation concentration could provide an opportunity to further reduce the load on the top level server. For instance, an intermediate level server could be placed at United States Air Forces in Europe (USAFE) to concentrate all of the traffic from the geographic area and minimize the amount of traffic making transoceanic voyages. To the local servers the intermediate level would look like the top level server. But to the top level server the intermediate level would look like a local level server.

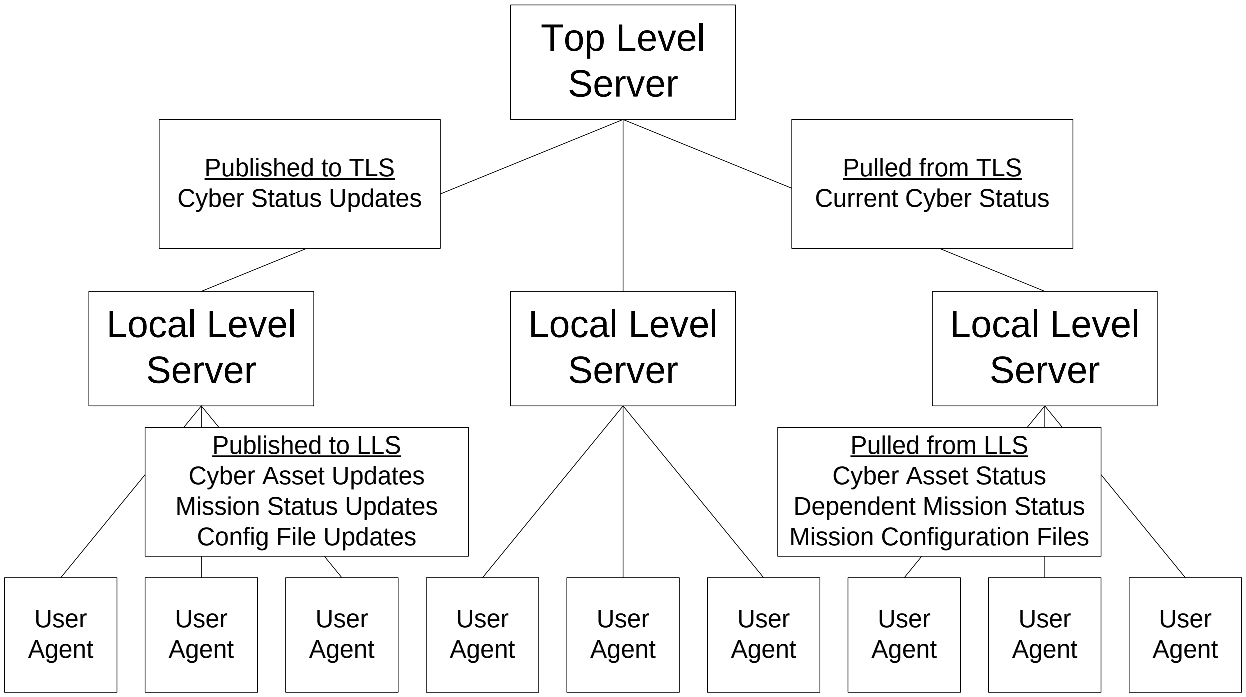


Figure 6. Multi-base, multi-mission view of the proposed architecture.

In a hypothetical scenario where USAFE had its own intermediate level server, all of the bases on the continent use the also hypothetical European Gadget Organizer (EGO), a cyber-based system used only in USAFE for tracking hand-held devices that did not meet a threshold for inclusion elsewhere and only had to be tracked in USAFE due to congressional mandate. EGO qualifies as a cyber-based system and resides on Ramstein AB’s computer network. Without an intermediate level server, EGO would have to report its status to the top level server when there was an incident, and all of the local level servers in USAFE would have to send requests to the top level server in an interval commensurate with the most needy of the missions on their particular base.

If instead there was an intermediate level server in USAFE, EGO would report its status to the local level server when there was a status change or incident. Those updates would transit through the local level server to the intermediate server and out to the top level server. Requests for EGO's status would first go from the local level servers to the USAFE intermediate level server. If the cached data was current enough for the request, the intermediate server would return that status without making an additional request to the top level server. If not current enough, then the intermediate server would request a more current status. So rather than *n* bases asking for status over transoceanic circuits, instead the request would only be made when the data was too old, reducing the strain to approaching 1/*n* of the original volume on both the top level server and the associated network paths.

3.3 Primary Investigative Question 2: How would this architecture support the receipt of mission-relevant notifications?

The key to this architecture is that it returns communications personnel to the role of providing a utility rather than trying to perform mission assessment for the operations community. Providing mission owners become responsible for publishing their status. Dependent mission owners identify what it is that is relevant to their mission and pull those statuses in an interval commensurate with the perceived level of importance to their mission.

By stripping the intermediate steps in the notification process, any remaining inadvertent filtering and noise are self-generated. Providing mission owners determine what notifications to provide to downstream users. These notifications can be augmented by network security personnel if they are able to determine there is an outage or some other form of problem. Dependent mission owners determine what notifications they want to receive. Noise enters the system only when false information is transmitted by the providing mission owner. Noise is received by dependent mission owners only when bad information has entered the system or when a subscription is not relevant to mission accomplishment. Inadvertent filtering works in the same way in which it may only be implemented by providing system owner by choosing not to publish a status or by the dependent mission owner by choosing not to subscribe to the needed resource.

The user agent acts as the control panel for each enduring mission. Publish actions are done through the agent and are sent to the local level server. A subscription to a particular mission notification is little more than storing the identification of a needed resource and the time interval at which the resource should be checked. That subscription is stored in the mission configuration file for the dependent mission.

The user configuration file is a cornerstone of this architecture. It allows those that share the authority for a mission to have a unified view of what provider missions are publishing as their status. This enhances Level 1 EMSA because all are seeing the same published information [35]. That shared common picture extends to both those in leadership positions of the particular enduring mission as well as to the applicable echelon users from whom their authority was granted. As an example, the configuration file for 85 CS/SCOI can be viewed and monitored for alerts by 85 CS/SCO—85 CS/SCO was the mission from which the authority for SCOI originated. For non-hierarchical enduring missions, the party from which the delegation of authority and communication of commander's intent originated would be the first person in the hierarchical chain who could view that configuration file.

The user agent is a very rudimentary decision support system. The user agent does not make any decisions for the dependent mission stake holder. It only receives notifications and provides an avenue of Level 1 EMSA for the dependent mission stake holders [35]. It also provides a conduit for passing similar notifications onto downstream users. However, as designed it is only as smart and thorough as the inputs that are provided to it. This presents an excellent trail for future research to take efforts like which have been done by Milcord on Commander's Learning Agent (CLearn) [49]. In their system, desktop interaction with external systems is observed and CLearn recommends additional systems for monitoring to the user that may be vital to mission accomplishment. Figure 7 shows a block diagram illustrating basic data paths for CLearn.

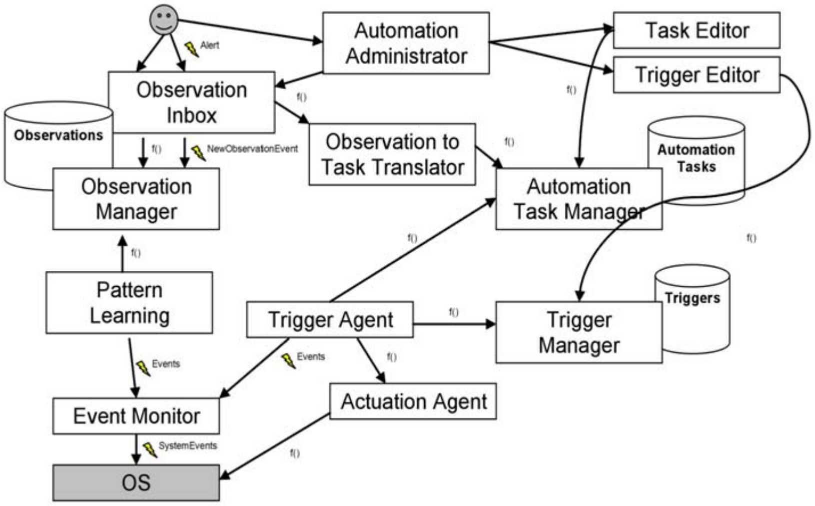


Figure 7. A basic diagram of CLearn from Milcord [49]

Additionally, the decision to then use the user agent and make an announcement to all downstream users to inform them that an issue has been identified that may affect their missions also requires human intervention. In this implementation there is no mechanism to receive a notification from an upstream provider and have the agent automatically send a warning. Again, this is a rudimentary decision support system--it needs human interaction to move from Level 1 to Levels 2 or 3 EMSA [35]. This also presents an excellent opportunity for future research to either implement a rules-based mission capability scheme or a case-based learning scheme in which when notifications are received they can be compared against previous actions and a notification can be sent to downstream users autonomously. Rules-based mission capability will be discussed later in this chapter.

What sets this architecture apart from many other systems is the ability for more than one user to receive the alerts on the same mission. Multiple decision makers have concurrent access to the same configuration file. This severely mitigates the potential that an alert will be sent out but will not be received by its intended recipients. In a fictional work environment as shown in Figure 8 and Figure 9 below where three personnel are authorized to load the configuration file, any or all of the three personnel may be monitoring that configuration file at the same time. Provided one of the three personnel is present for duty, then the alert has the potential to be received. If more than one is present then all present will see the same information at more or less the same time. If none are present for duty, then when they return then the alert will be waiting to be received upon their return.

To take into account the presence of additional duties that use upstream missions, an individual user has the ability to see multiple configuration files. Users in the same hierarchical level may have different sets of configuration files loaded in their user agent, depending on both their role in that hierarchical level but also additional duties that they may be responsible for.

There is a separation between the person, who may have the authority to execute more than one portion of the commander's intent, and the individual missions for which authority has been delegated to. The user-level agent is the means to create and modify configuration files for each enduring mission. During initial setup, a leader can identify what it is that they use and set an alert commensurate with the importance to the mission. As the mission evolves and the available tools change, modifications can be made to the user configuration files.

In Figure 8, three individuals work in a fictional SCOI workcenter. All three have authority to act for the SCOI mission. Individual A has an additional non-hierarchical duty as the unit safety monitor and reports directly to the commander. Indivdual B has an additional non-hierarchical duty as the unit physical training (PT) manager. All three individuals have SCOI's mission configuration file loaded to their user configuration file and can receive alerts on those missions and assets they depend on. However, an alert for the safety additional duty will only be received by individual A because that individual is the only one that has safety loaded to their user configuration file and has had commander's intent communicated.

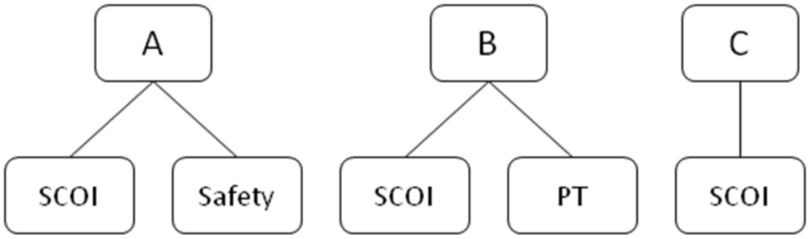


Figure 8. SCOI Workcenter User Configuration Files

Figure 9 represents a similar scenario. Again, three individuals in the same workcenter. All three individuals have been granted authority for the primary mission (Mission A). Two of the individuals have been granted authority for non-hierarchical missions (Additional Duties B and C respectively). When there is a change to status for Mission A, all three get an alert in their user agent. When there is an alert for Additional Duty B, only individual A gets the alert because only individual A has authority for that mission and can act on the commander's behalf to resolve it.

Meanwhile, though not shown, the squadron commander has visibility of both of the alerts, received as the first person in the echelon for the additional duty, and *n*th person in the echelon for the main mission after passing hierarchically through all the previous levels.

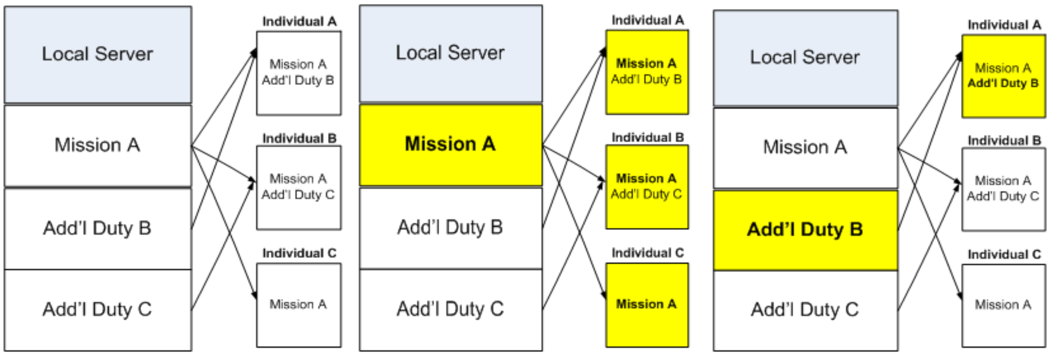


Figure 9. User configuration files and alerts for Mission A and Additional Duty B

One of the strengths of this architecture is the use of these configuration files and having them available to multiple users. Having this configuration file is the equivalent of having a virtual continuity binder. When circumstances dictate that there is a change in leadership at a particular enduring mission, providing the new leader a picture of what missions s/he is dependent on is vital to minimizing the learning curve. Having these configuration files present means that there is no guess work as to what the mission depends on and no guess work as to who to contact when something goes horribly wrong. What is missing is an understanding of why it is that losing a particular mission dependency is harmful to mission execution. This is not particularly a technology problem but more a communication of commander's intent and mission application problem—one that is outside the scope of this research and is a problem that leaders at all enduring missions struggle with when they receive and/or relinquish the reins of leadership.

It must be stressed that the user-level agent must also publish changes to mission status for the architecture to meet its mission of proving timely and relevant notifications.

Downstream missions rely on the enduring mission being monitored by the user-level agent. The same user-level agent that receives the notifications is the one that publishes the notifications for others. If, for example, a cyber resource mission provider knows that there is a problem but fails to publish that notification (and if the base or top-level servers do not perform a manual update), then none of the missions dependent on that resource will know that there is a problem. There is still a requirement that a human somewhere in the process start the notification process. This is not unique for this architecture, but the onus relies directly with the owner of the providing mission.

3.3.1 Investigative Question: How does the mission-level agent pull statuses?

Those who have modification access to the user-level configuration file must identify what the mission dependencies are. It is proposed that this is done with a subscription model utilizing an identification code (which later will be called a Mission or Asset Identification Number) that is unique to that mission being provided. Each local level server would have its own identifier prefix to identify those missions that are held by that server. One way this could be done is through the use of a base-unique identifier followed by a unit designation and then a unique multi-character code.

To illustrate this, at the fictional 85th Communications Squadron at Keflavik AB, Iceland, one of the major cyber assets is the local electronic mail server. When this asset is put online, the asset needs to have a unique identifier. To give this asset an identifier, the following would be combined to form its mnemonic.

- **IS** for the base (same as the “tail flash” of the fighter aircraft stationed there),
- **COM** would identify the unit, and
- a 3 character identifier (**6YA**)

These would be concatenated to form ISCOM6YA. Under this scheme, each unit would have a namespace of 36^3 or over 46,000 possible mission identifiers. The 3 character identifier could be issued sequentially (numbers and then letters), randomly to provide a small bit of obfuscation, or with a set scheme as determined by the unit. The local level server would keep track of these identifiers and humans would manually add them to the server. Because this is a cyber-based asset, when the identifier is established its existence is published to the top level server by the local level server.

Asset or mission owners could publish this code openly or require that to obtain the code a mission stakeholder must perform some form of request. Such a request could take any number variations (i.e. e-mail, phone call, web-based form, ad nauseam) depending on any number of factors as determined by the providing mission.

Those who wanted access to the statuses would have to know the code and enter it into their configuration file. This would be done within the user-agent and would require the code and the time interval to pull the requests from the local level server. In cases where there is a desire to further lock down the process of obtaining updates, a mechanism could be placed in the sign-up process that grant the providing mission or asset owner the ability to approve or disapprove an add request.

Once the addition was saved to the configuration file, all users of the configuration file would receive notifications when there was a status change. To receive the change to the configuration file, the user-level agent would also periodically check to see that the current mission configuration file is the same as the file that was loaded into their user agent. If there is a change, the new configuration file is loaded and acted upon.

In operation, the user-level agent sends out a request to the local level server on the time interval saved in the mission configuration file. The request that is sent contains the resource identifier and how recent the last update needs to be. If the request is a local level resource, the how recent field is ignored--it is assumed that the latest update on the local server is the latest update put forward by the providing mission owner. If the request is for a resource at a different base, the local level server checks the timestamp of the cached data against the current time and how recent of an update is required. If the cached data has been retrieved recently enough to meet the needs to the requesting mission (or is fresh enough), the local level server responds to the requesting mission with the cached data. If instead the timestamp of the cached data is earlier than is acceptable for currency based on the user agent's requirements (or more simply put, is too old or stale), the local level server sends a request to the top-level server. When received, the local level server time stamps the newest data and then completes the request to the user agent.

In this example, the mission stakeholders responsible ISCOM6YA have identified three key items necessary for the mail server to work: The Domain Name Server (DNS) server, the commercial circuit leaving the base, and a mail gateway back in the states.

Each one of these cyber assets would have their own identifier. The mission stakeholders have obtained the identifiers for its dependent missions and they enter those codes into their user agent configuration file and provide time intervals to check those assets. On a regular basis consistent with the identified time interval, the user agent contacts the local level server. The local level server responds (or squawks back) the last received status from the two local assets because the assumption exists that unless a status update has been published that the last published status is the most current. For the external asset the local level server checks time stamp of the most recently cached status. If the timestamp of the cached status is stale, the local level server contacts the top level server for an update. The top level server responds with the most recently published status. The base-level time stamps that status and then forwards it back to the requesting user agent. This process continues until such time that the user agent is turned off.

To further illustrate this in Figure 10, a mission stakeholder for ISCOM6YA (TSgt Adams) returns from lunch at 1300 and turns on their user agent. Upon initial startup, the configuration file is loaded, and an initial set of checks occur. The base server receives requests from the user agent to provide the most current status for the three monitored resources.

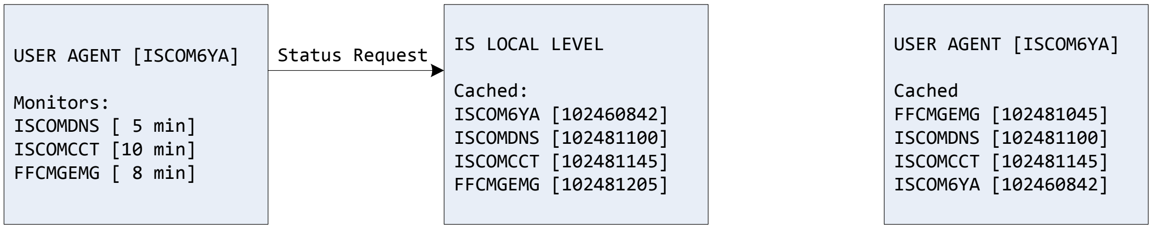


Figure 10. A status request is made from the user agent to the local level server.

The base level server immediately responds with the cached status for ISCOMDNS, ISCOMCCT, and ISCOM6YA as shown in Figure 11. These are local resources and it is assumed that the cached status is the most current status. The timestamp for the cached copy of FFCMGEMG's status (the e-mail gateway at Langley AFB) is checked against the requirement for the information to be only 8 minutes old. As it is now 1300Z and the last cached data was timestamped at 1205Z, the base level server needs to refresh its data. This prompts the base level server to request a new status from the top level server.

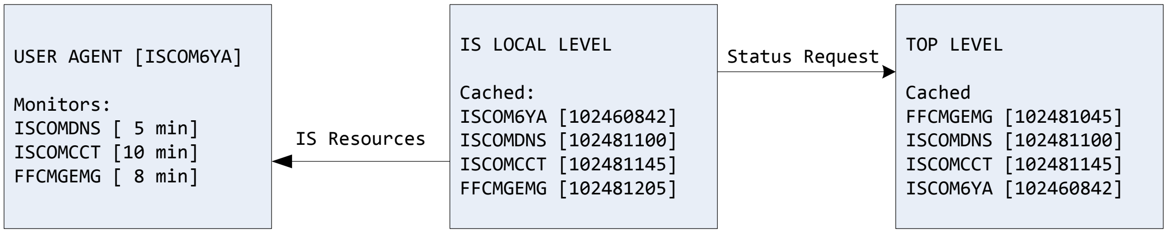


Figure 11. Local server responds with the status of local resources, inquiries top level server

In Figure 12, the top-level server received the request for the status for FFCMGEMG and responds with the current cached status. The base level server receives the update, timestamps the update with the current time, and sends the update to the ISCOM6YA user agent.

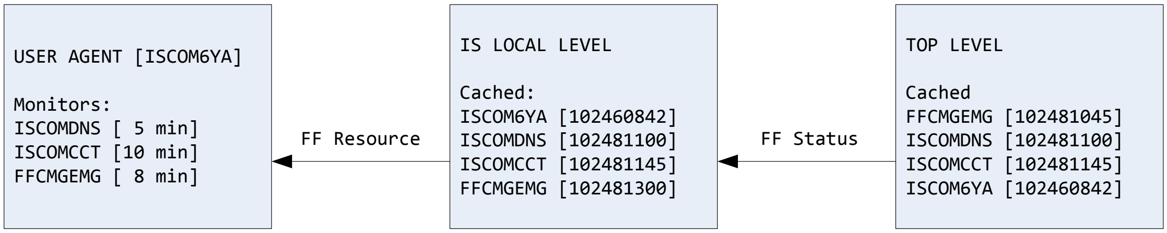


Figure 12. Top level server responds with cached status, local level timestamps status and sends to user agent

Upon receipt of the initial requests, the user agent would display to the user the current status of the assets that s/he cares about. After the initial requests are made, the new requests are made at the time interval specified when the configuration file was created and/or last updated. This means at 1305Z the user agent will request the status of ISCOMDNS, at 1308Z the user agent will request the status of FFCMGEMG, and at 1310Z the user agent will request the status of ISCOMCCT. The user agent would only provide an alert if there was a change to the status.

At 1530Z that same day, a disk corruption issue happens on the e-mail server. There are numerous missions that are dependent on the e-mail server being up and the commander has communicated her intent that when there is a status update that it needs to be published via the user agent. The person with the agent running updates the status and publishes it. The base level server receives the update and because when the identifier was set up that it was annotated as being a cyber asset, that status is also published to the top level server. Figure 13 shows this process in that an update is sent by the user agent to the local level server. The local level server stores this information so that other local level organizations can see it.

The local level server also publishes this information to the top level server. This publication to the top level server allows both external entities that rely on the exchange server (which admittedly should be few) to see the status update. It also provides the potential for top level server administrators to see this failure. Should there be an instance where multiple bases all have the same problem at the same time, this provides the potential for top echelon network administrators to see this trend and determine if it is an anomaly or a wide-scale attack or software failure.

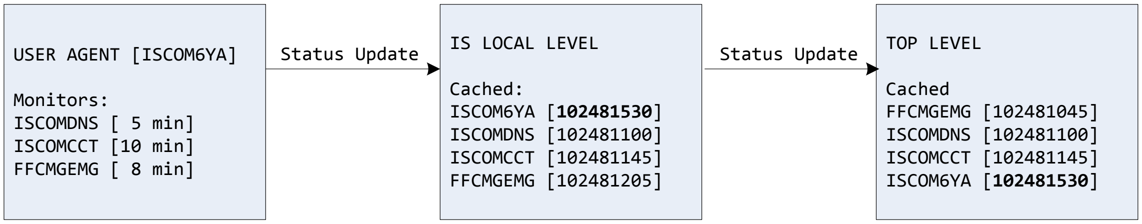


Figure 13. Cyber asset ISCOM6YA fails. User agent sends update to local server which is echoed to top level server**.**

Later in the day, TSgt Baker comes in to work second shift and starts up the user agent on her system. Like before, the user agent pulls down the current statuses of the monitored resources. The only difference is that because there are two user agents both asking for the status of FFCMEMG, the cached data will be younger than 8 minutes for TSgt Baker so an additional request to the Top Level server will not be required. When TSgt Adams logs off of his instance of the user agent, then TSgt Baker will be the only one requesting status and it will be her requests that spurs the request to the top level server. There is no specific limit of how many individuals can be running the user agent with this mission identifier loaded at any given time.

Provided in this example is a three level notification system which illustrates that it is the minimum to achieve Level 1 EMSA on and between bases. As described before, additional layers could be placed between the top level server and the base level server to help distribute traffic as required. Logic seems to dictate that doing so in a similar construct as is found above the base level in AFI 38-101 (from HAF to NAF levels) could be one workable scheme [19], but such decisions are left to be made by those who attempt to implement this architecture.

3.3.2 Investigative Question: How does the user-level agent become relevant to enduring missions?

The user-level agent becomes relevant to enduring missions because it both provides Level 1 EMSA to those watching the agent and provides a conduit to communicate Level 2 EMSA to dependent missions.

The individual monitoring the user agent now knows as much about the missions and assets he or she depends on as the providing mission owner is willing to share. With that information combined with the other assets that go into the mission (i.e. tools, people, training, supplies, ad nauseam), the person with the delegated authority can determine the mission capability to perform the mission(s) delegated to them by the authority of the commander and to the level as desired through the stated commander's intent.

Mission capability does not equate to mission success nor does it equate to mission failure. Mission capability only states whether or not the conditions are present that will make it likely that given what is on hand and successful execution of commander's intent that the mission will be carried out successfully. Human nature remains human nature. There are few guarantees that everything will go according to the notional plan.

Measuring mission capability is a continual measurement. At any given time there is a readily identifiable benefit to a real-time mission capability status of an aircraft. It is safe to say that when boarding an aircraft that most individuals (whether passengers or crew members on the manifest) would like the aircraft to be FMC. It is also safe to say that when operating or depending on any weapon system, knowing if it is capable of carrying out its mission would be information a commander would want to know.

Measuring nominal mission capability can and has been extended beyond the JP 1-02 definitions of physical objects and into more abstract constructs. Air Sovereignty missions are an example where mission capability has been extended to an abstract construct of mission. JP 1-02 defines an Air Sovereignty mission as, "The integrated tasks of surveillance and control, the execution of which enforces a nation's authority over its territorial airspace."

In addition to this extension of mission capability to the abstract, Air Sovereignty missions are also an example of an enduring mission that have employed real-time rules-based mission capability monitoring. The Iceland Air Defense System (IADS) Maintenance Control Operating Instruction (OI) illustrates how this can function [16].

The IADS, formerly a mission of the United States Air Force's 932nd Air Control Squadron, was tasked with maintaining watch over Iceland's Air Sovereignty. Within the

OI was a rules-based system used to determine nominal mission capability of the overall mission. An exhaustive list of key parameters had to be met in order for the IADS to be considered FMC. Each of these parameters had thresholds by which the nominal mission capability of the mission was declared. A parameter dropping below a specified number could render the overall mission PMC or NMC, depending directly on how far away it deviated from the standards [16].

When FMC could not be achieved, those in charge of the IADS had to contact those who relied on IADS to inform them of the status. Those dependent missions then had to determine what effect it had on their mission capability, often with their own set of rules.

Outside of these mature and reasonably static systems, mission capability and mission impact in the face of a cyber incident is performed manually. The leaders receiving these updates have to determine what this loss of a resource will mean to the mission. If a resource is absent or compromised, is there impact to the ability to execute the mission? This is where Levels 2 and 3 of EMSA apply [35]. The strength of the leader and their understanding of how something will impact their mission will affect the accuracy of the resulting messages that they squawk regarding their mission capability.

3.3.3 Primary Investigative Question 3: What details are missing?

As stated before, this architecture and accompanying user agent is a very a rudimentary decision support system. It provides only Level 1 EMSA and only to the extent to which everybody is willing to share information and to the extent that mission stakeholders are willing to identify the correct providing mission dependencies and monitor the agent. It does not replace the human mind in making decisions when it comes to what to do next.

There are a number of additional basic things that would have to be settled to make this architecture function. The primary issue would be data format of the status messages. This is both a mission issue and a technology issue. A very simple approach would involve just a small number of fields to include time stamp, narrative of the status, warning flag, and some form of message authentication. This could be done in the form of a text file, an XML file, or built into a database.

Identifiers for the missions being monitored is another item that would need to have the details worked on. The example above was provided for illustrative purposes only. It may or not meet the actual operational needs.

Bringing this to a joint environment would provide its own set of issues and hurdles. One possible solution for this could be to establish a server at a level above the top level server and this server would act as a clearing house between the services and the unified commands.

Finally there is again the issue of OPSEC as defined by JP 1-02 and discussed earlier in Chapter I. This much information in one place could create a treasure trove of information that would make our enemies desirous of cracking it and would cause lost sleep for information security officers. The proper network and classification to attach to this architecture is something left for those experts [3].

IV. Methodology

4.1. Overriding considerations

This architecture is designed to provide a connection between mission and asset owners and dependent/downstream users to allow for instant notifications of cyber incidents. It is also designed to allow those dependent/downstream users to evaluate what that incident notification means to them and provide warnings to their downstream users if they are not able to perform to full mission capability. The goal is Level 1 EMSA [35] for dependent/downstream users.

In planning for this architecture, a top consideration was for the end product to be usable by a wide range of mission stakeholders. The target audience for this architecture and the interface to it could reach the ranks of E-3 (Airman First Class) and O-1 (2nd Lieutenant) on the youth end of the spectrum, to E-9 (Chief Master Sergeant) and O-7 (Brigadier General) on the experience end of the spectrum.

Of chief concern as well is that any solution should reduce the administrative burden on communications professionals, not add to it. As such, considerations were made while designing methods so the menu of available actions for administrators was purposefully small. Network administrators should only have to register users into the system, process paperwork, and override mission status when the network administrators (whether locally or up the network echelons) can see a problem that the system or mission owner may not be able to see.

4.2 Pull

The decision to implement this as a pull form of communications rather than a push was based on the fact that both AFI 33-138 [2] and most every other scheme explored during this research has relied on push rather than pull. The notable exception was C2RMS, which for availability checking was pulling in the form of pings or inquiries. C2RMS was a product that had been used successfully outside of the laboratory environment for the purpose that was intended [32]. This lent credence to the concept of pull.

4.3 Scalability

Scalability was also a chief design consideration. While networks are more robust and carry far more bandwidth today than they did ten years ago, enough drops of rain are enough to flood a village.

As design considerations were drawn and discarded, certain themes kept recurring. They included:

- Keep data messages compact yet relevant
- Check only as often as necessary
- If cached data can be used to reduce the burden of network traffic yet still be current enough to meet mission needs, then use cached data.
- Balance the load where possible by allowing the potential for intermediate servers to keep local servers from inundating the top level server

- Allow the top echelon of network users as rich of a data set as possible for the equipment and services that matter to them, yet exclude what is not important.

4.4 Workflow modeling decision

In Chapter II there was a discussion regarding workflow modeling methods. Based on the relevant research and a desire to bring formality to this research, it was acknowledged and embraced that to demonstrate the usefulness of the architecture proposed would require formal modeling.

This investigation began with Department of Defense Architecture Framework (DoDAF). DoDAF is the Department of Defense standard. Shaw, in his research on modeling mission impact analysis on network outages, used DoDAF to show the interconnections within a CAOC [50]. DoDAF is a very robust modeling language. But as the research started pointing towards an object-oriented environment it was quickly apparent that DoDAF would not be an appropriate way to demonstrate the concepts at hand [36].

Also reviewed but discarded was YAWL. YAWL, a branch of Petri nets, was developed, "…to allow for a more direct and intuitive support of the workflow patterns…" as compared to that which was found in Petri nets. The biggest limitation to this research discovered when working with YAWL was that YAWL seemed to be more shaped for processes with a definite beginning and end. For discrete missions this would have worked well, but enduring missions are by their definition closer to an infinite loop than a definite start and finishing [38].

Also considered but eventually not used were the IDEF series of languages. IDEF initially seemed to have the most promise because it easily demonstrated the concept that a mission consisted of the products of other missions, people, and rules providing a product that in turn could be provided to other missions. This path was also evaluated and eventually set aside because of the linear nature of the language and not being as friendly to an object oriented approach [39].

While not the last thing investigated and vetted, Unified Modeling Language (UML) version 2.0 became the standard used for the designing of this architecture. The biggest strength of UML was that it was object-oriented. From very early in the research the visualization kept pointing towards each mission as being not much more than objected-oriented programming (OOP) classes—they import things, they have rules they must follow, they use processing power, and they return something. While more complicated than simple OOP exercises, the concept remained basically the same [41].

UML also provided different levels of granularity in drawings. This allowed for top-level drawings that integrated with lower-level drawings and provided the level of detail needed for each particular stage in the research. As UML is an industry and academic standard, it provided structure and credence to the ongoing research, yet had enough flexibility that new concepts and novel approaches to problems could be entertained and explored.

Enterprise Architect was the engine chosen to create the drawings. This was chosen after Visio was tried but quickly discarded because Microsoft has made the decision to not to support UML 2 in lieu of a different standard. Enterprise Architect stayed close enough to the standard to provide accurate and usable drawings for the research.

4.5 UML and Systems Analysis

UML was used in order to bring formality to designing the building blocks of this notification architecture. Dennis et al make this modeling the foundation of the analysis portion of the software development lifecycle. The first three parts of the analysis process are gaining a comprehension of what the current system is, how it can be improved, and developing a course of action for the improved system [41]. These steps were performed during the course of the research and the results thereof are in Chapter II and Chapter III of this report.

The next three steps in the process involve designing use cases and then fleshing out the use cases with structural and behavioral models. The use cases should represent how the customers are going to use the system. The structural and behavioral models will then provide basis for the data structures and algorithmic design that will go into creating the new system. If the customers have fully identified their requirements and are able to effectively communicate those requirements to the software engineers in the initial steps, then the design process is hastened by diminishing the need to redesign the system to account for missing features or functionality. In the current theory of systems analysis and design, these steps should be performed before a single line of programming code is written [41].

This research relies on five different types of UML 2 drawings to demonstrate the proposed architecture, supplemented sparingly with traditional flowcharts where additional detail is desired. These five are a Use Case diagram, a Class diagram, Sequence diagrams, Communication Diagrams, and Activity Diagrams.

The Use Case diagram is the highest level of the diagrams. It is designed to show what parties (known as actors) interact with the system and what the various use cases are [41]. Six primary use cases are spelled out in this research to account for the bare minimum of functionality required to meet the overall research goal of providing timely and relevant notifications to mission stakeholders.

The Class diagram is a structural diagram. It contains structures for key roles in the architecture. Each key role has an area for things that role is (nouns) and things that the role can do (verbs). It is this class diagram that acts as a basis for the eventual object oriented programming (OOP) in which objects (also known as classes) are built from [41]. There can be multiple instances of an object, and this is accounted for in the drawing as appropriate.

Sequence, communication, and activity diagrams are all behavioral diagrams. Sequence diagrams are the most general and show the actors that are involved in the particular use. Communication diagrams are more detailed, but are more process-centric than actor-centric. Activity diagrams have the highest level of detail and combine many of the aspects of sequence and communication diagrams. All three diagrams are used as aids for algorithmic design [41].

With these steps completed, the next logical step is to begin the process of coding the system. This next step falls outside of the scope of this research and is left for future researchers to tackle.

V. Use Cases, Structural Models, and Behavioral Models

5.1 Purpose and Introduction

The purpose of this chapter is to document the results of the research and the resultant use cases, structural models, and behavioral models proposed to meet the goal of timely and relevant notification of cyber incidents to mission stakeholders so they may assess the mission impact of the incidents.

The means of demonstrating how the proposed architecture would operate is through these structures as developed under the guidelines of the UML version 2 standards with variations as authorized under the standard. The reviewed references [41] [42] have pointed out that there is room for flexibility in the standard and this research has taken advantage of that flexibility when needed to illustrate points that could not otherwise be easily accomplished.

5.2 Definitions

This research will fall back to the baseline of terms and entities contained in AFI 33-138 [2]. It is acknowledged that the constructs of the Air Force enterprise today is significantly different than when the instruction was written and approved. While the names and functions of most of the intermediate steps and offices have changed in the five years since publication of AFI 33-138, the entities of the greatest importance for this research are:

- The mission stakeholder
- The local base-level communications entity

- The top entity in the Air Force network enterprise.

There still exists a hierarchical chain of responsibility from the user up to the top of the enterprise regardless of what the current names are.[4] On the first page of all AFI's is the statement, "Compliance with this instruction is mandatory," and accordingly this report will maintain uniformity of terms with the instruction that is in effect at the time of publication of this report.

5.2.1 User

A user is any human or entity who is authorized to use this architecture system. Users must have an account on the network on which the system is located. Using the Air Force as a baseline, a person who is a user must have a Common Access Card (CAC), have filled out the requisite paperwork to obtain network access, have a security clearance commensurate with classification level of the system that they are requesting access to, have a validated need to be on the network, and have been authorized an account. Non-human entities are anticipated as the technology and theory increases. There are four typical variations of user in this architecture and a user may be any combination of these variations to include no variation at all.

5.2.1.1 Administrator

An administrator (or admin) is someone with rights to add and remove users and their configuration files, add and remove missions and assets status files, update mission

[4] For example, during the course of this research both Air Force Cyber Command (AFCYBER) and 24th Air Force were both established and operationalized, changing names of the entities that were in conflict with the established instruction.

and asset statuses, and add or remove mission or asset responsibility from a user. Their involvement in the system on a minute-to-minute basis should be minimal.

5.2.1.2 Stakeholder

A mission or asset stakeholder is an individual who has authority for a mission or asset. This authority is an extension of commander's intent as defined in JP 1-02 [3]. Within the context of this architecture, an individual assigned as a stakeholder for a particular mission or asset can add or subtract missions and assets to be monitored for the accomplishment of the mission, view the status of monitored missions or assets, and update the status of their assigned mission or asset. A mission or asset stakeholder may delegate their authority so that those who work for them may also perform all of those tasks. The mission or asset stakeholder will receive alerts through the architecture based on the missions or assets that they have chosen to monitor. That stakeholder may then update the status of their asset or mission based on the status of the missions or assets they are monitoring and depend on to accomplish their assigned mission or make their assigned asset available.

A stakeholder's function in this architecture is to transform the Level 1 EMSA received through the notifications into Level 2 EMSA so as to correctly interpret what it means right now to their ability to provide the mission or asset to others. Ideally, with a well-informed stakeholder, they can extend that Level 2 EMSA into Level 3 EMSA and project what is likely to happen next to give downstream consumers a preview of what could or is going to happen so that they may plan accordingly [35].

5.2.1.3 Echelon

Typically a stakeholder is someone who performs the day to day actions to ensure mission accomplishment or asset availability, but a stakeholder may also be the person who delegated the authority for that asset or mission. This third sub-type of user, an echelon user, has organizational authority over the mission and asset stakeholders. They have the ability to not only monitor these missions and assets as the stakeholder could, but also drill down into the details to gain a broader perspective of the current situation. They may also assume the role of the stakeholder to exert control over what is being advertised about that mission or asset.

5.2.1.4 Viewer

The last subtype of user is a mission or asset viewer. This type of access is granted to those personnel who are authorized to monitor the health of the mission or asset, but has insufficient knowledge, training, or rank to be trusted with being able to speak for the mission or asset stakeholders on the Level 2 or Level 3 EMSA for that mission or asset [35]. These personnel report to a stakeholder or echelon user who can perform the necessary updates.

5.2.3 Mission

Mission was covered to great depth in Chapter II, but for the sake of repetition and as it applies in the architecture, it is a task or service which benefits others. This includes individuals who may not want to benefit from the task or service (i.e. targets to be bombed, monitored, or infiltrated)[5]. The granularity of task or service is dependent on the situation and who it is that needs the task or service. There is a direct to geometric proportional relationship between the fineness in granularity of mission monitoring and the amount of effort required to update and monitor the missions. Missions tend to be abstract in nature.

5.2.4 Asset

Assets are a super class of missions. Assets are physical or abstract constructs that another mission or asset depends on. Examples of physical assets would include but are not limited to people, vehicles, paper, fuel, telecommunication cables, and servers. Abstract assets include data and handshaking passing on a wire, power, cold air, and so on. Assets may be consumable (i.e. power, fuel, or water). Assets may be damaged and may or may not be repairable. Assets may be temporal. Assets, like Russian nesting or matryoshka dolls, may be multi-layered and each layer could be monitored separately. All assets can be monitored for their availability. Cyber assets can also be monitored for their confidentiality and integrity.

5.2.5 Cyber Asset

Cyber assets are assets that reside on or in the cyber domain [34]. This includes but is not limited to:

- Infrastructure such as long-haul circuits, switches, and routers

[5] It is somewhat taken for granted that this type of recipient will not be given access to the mission or asset status. In this type of instance the idea is to deny Level 1 EMSA from the recipient.

- Servers
- Operating systems to include proxies and intrusion detection systems
- Applications to include e-mail server software and database software
- Data contained inside the applications

Information is specifically not listed as an asset. Data is Level 1 EMSA. It doesn't become information until a human interprets it which turns it into Level 2 EMSA or projects what it will mean in the future, making it Level 3 EMSA. The proposed architecture is designed to provide only Level 1 EMSA [35].

5.2.6 Dependency

A dependency is any mission or asset that another mission or asset relies on to contribute to mission accomplishment. Not having a dependency available does not necessarily mean that the mission or asset that relies on it will fail. It does mean that one tool in the virtual tool box of that reliant mission or asset is not present. The dependency's importance to the reliant mission or asset is temporal—that its measure of importance depends on what other factors are present at any given point in time. Chapter III discussed how, in mature and reasonably static environments, there can be rule-based measures that will dictate what the impact of the loss of particular dependency is. In most other situations, knowledge of the loss of that dependency provides, at best, Level 1 EMSA and it is up to knowledge of the other component parts of the mission or asset to determine Level 2 or Level 3 EMSA [35].

5.2.7 Notification Systems

The local notification system (LNS) is a server within an installation's local, metropolitan, or wide area network that holds user configuration files (UCF), mission status files (MSF), and asset status files (ASF). Asset files contained on the installation which are coded as cyber assets are published from the LNS to a top-level notification system (TLNS). The status of cyber assets which do not reside within the local installation but are dependencies for local missions or assets are pulled from the TLNS and cached locally on the LNS to minimize traffic to the TLNS.

As the name would imply, the TLNS is located at the highest echelon of Air Force Network Operations (AFNETOPS[6]). All assets coded as cyber assets have their statuses stored on this server. There are three primary reasons for setting things up in this manner. The first is so that AFNETOPS personnel can potentially mine the data contained within these files to determine trends and on-going attacks. The second is that AFNETOPS personnel will have the best overall picture of all of the threats to the network and those personnel can override what is contained on the a cyber coded asset file. And last, by keeping the files in a central location that all LNS's access, it minimizes what an attacker can glean from seeing who is connecting to what bases to get status files. Connections made from LNS to LNS run the risk of revealing to an attacker where the crown jewels are based on network traffic and where to attack. Centralizing

[6] Again, AFNETOPS is a legacy term from AFI 33-138 as published. The name, concept, and who is responsible for this entity has changed at least once during the course of the research and very well could change again prior to publication.

this traffic to a single server obfuscates who is going where for a specific type of resource and places all of the data behind the strong walls of the top of the AF enterprise.

As discussed in Chapter III, there may be instances where it would be helpful to add levels of hierarchy between the LNS and TLNS. As described, an example of this would be geographic considerations such as forces stationed in Europe. Rather than having multiple bases going all the way across the Atlantic to get requests and send them back it may be quicker and more efficient to have an intermediate level notification system (ILNS) act as a cache for status files of assets coded as cyber for those systems being used in the region. With an ILNS, all of the installations who use a particular cyber asset can check the ILNS first. If the data is fresh enough to be usable, then there is no need to for that transoceanic request. But if not, then the request is made, is stored locally at the ILNS, and then is forwarded on to the requesting base. How to establish when to make that kind of a decision and the mechanisms involved is purposefully omitted from this thesis as it falls outside of the scope.

5.3 Use Case Development

This research concentrates on six specific use cases. These use cases are what would be the bare minimum necessary to allow notifications of mission statuses to flow from providing mission or asset owners to those who are dependent on the mission or asset. It assumes that all mission or asset owners know all of their dependencies and/or missions or assets they want to monitor. It also assumes that these assignments are static.

These six use cases cover the six basic things the architecture needs to be able to do to make timely and relevant notifications. First, there must be a way to create users—the individuals who will be making and receiving the notifications. Next there must be a way to create the missions or assets to be monitored. Further there needs to be a way for users to be assigned authority for making notifications regarding incidents. There also must be a way to delegate that notification authority. Finally, there must be a way both to make that notification and for downstream users to receive that notification.

Some additional basic use cases can be enumerated quickly. As an example, for every use case that creates something a similar action should be present to delete that item. As stated in Chapter IV, the goal of the research is not to create a fully mature and operational architecture. Instead, the goal is to explore just those that are necessary to show how timely and relevant notifications can be passed.

These six use cases, along with those entities that interact with the use cases, are presented in Figure 14: the overall Use Case diagram. This diagram shows the highest level view of the proposed architecture. It will identify the actors, the six use cases, and the notional links between the actors and the use cases.

Next will be an overall Class diagram. As discussed in Chapter IV, the Class diagram identifies the major classes used in the notification system. Each class will contain both actions and attributes. Actions, as the name would imply, are things the classes can do. They are the verbs of the architecture. Attributes are the properties of the classes. They store the data that flows through the classes. They are the nouns of the architecture

Following the Use Case diagram and the Class diagram are the six use cases. For each use case there will be a minimum of three additional drawings. First will be a sequence diagram that will show the steps that occur within the use cases between the various actors involved. Next is a communication diagram that better mirrors the class diagram and the steps that occur to complete the use case. And finally is an activity diagram that resembles a traditional flow chart but adds the dimension of watching data move between actors. Where applicable, more traditional flow charts are also used to further delineate more complex concepts.

5.4. Overall Use Case Diagram and Class Diagram

The first step in describing this architecture is presenting the overall use case diagram. As shown in Figure 14, this diagram shows both the scope of the notification architecture along with the six use cases and their associated actors. An actor does not necessarily have to be human, as represented by the TLNS actor. Figure 15 shows the legend

Note that with all instances except for in registering a new user, it is a mission or asset stakeholder who is the primary initiator of the action involved. This is consistent with the stated proposal that the paradigm be shifted to mission or asset owners being proactive and pulling information regarding those things that they depend on rather than waiting for wing-level communications professionals to inform them that there is an incident to be concerned with. In specific cases other actors can initiate these actions (i.e. when an administrator sees evidence of an incident in progress), but these are the exception rather than the rule.

In addition to the format as specified in UML 2.0, these drawings are augmented with arrows pointing to parties who are involved. The choice to add those additional arrows was done because without the additional arrows it would be difficult to understand the scope and magnitude of both the architecture and the issues involved.

All humans in the use case are a form of a registered user. New users become registered users. Admins, stakeholders, and viewers are types of registered users. Admins have the authority to register users, designate individuals as stakeholders, or designate individuals as viewers. Stakeholders may designate registered users as viewers or stakeholders for missions or assets that they have authority for. Viewers may view the missions for which they have been grated authority but may not grant others any form of privilege. Registered users may do little more than authenticate into the system until they have been granted privileges by an admin or a stakeholder.

The non-human actors in the architecture are the LNS, ILNS, and TLNS. They do not initiate actions on their own nor do they necessarily benefit from any of the use cases. The LNS is involved in every use case. The ILNS and TLNS are only involved when the status of cyber assets are created, monitored, or modified for reasons discussed in Chapter III.

Each of the different types, colors, and thickness of lines have meaning. UML 2.0 does not specify colors or thicknesses and it is here that there is digression from and augmentation to the standard to increase readability of what is a busy diagram.

The thickest solid dark green line defines the boundary of the notional notification system. All six of the use cases fall within the notification system. All of the actors fall outside of the notification system.

Solid black lines without arrow heads are the benefitting actor. All of those solid black lines comes from the stakeholders. This is in line with the concept that the stakeholder is central to this architecture and are the ones initiating the bulk of actions.

Solid black lines with an arrow head shows a generalization. Admins, stakeholders, and viewers are a type of registered user. New users become registered users.

The solid light green line is used only once and it connects the viewer to monitoring for changes. The viewer does not initiate this communication per se, but instead is provided their authorization from either a stakeholder or an admin.

The dashed blue line shows data flows. Data flow can be bi-directional. Arrow heads at the end of the lines denotes that data flows in and out of the entity. A lack of an arrow head denotes traffic only out of the entity.

The thick dashed red lines show a participant in a use case that does not involve the flow of data and does not benefit from the use case. All but one of these instances is involvement with the admin. By definition the admin does not benefit from any of the actions. The registered user is the final instance of possessing such a line, and it is during the assignment of a registered user into a stakeholder or a viewer.

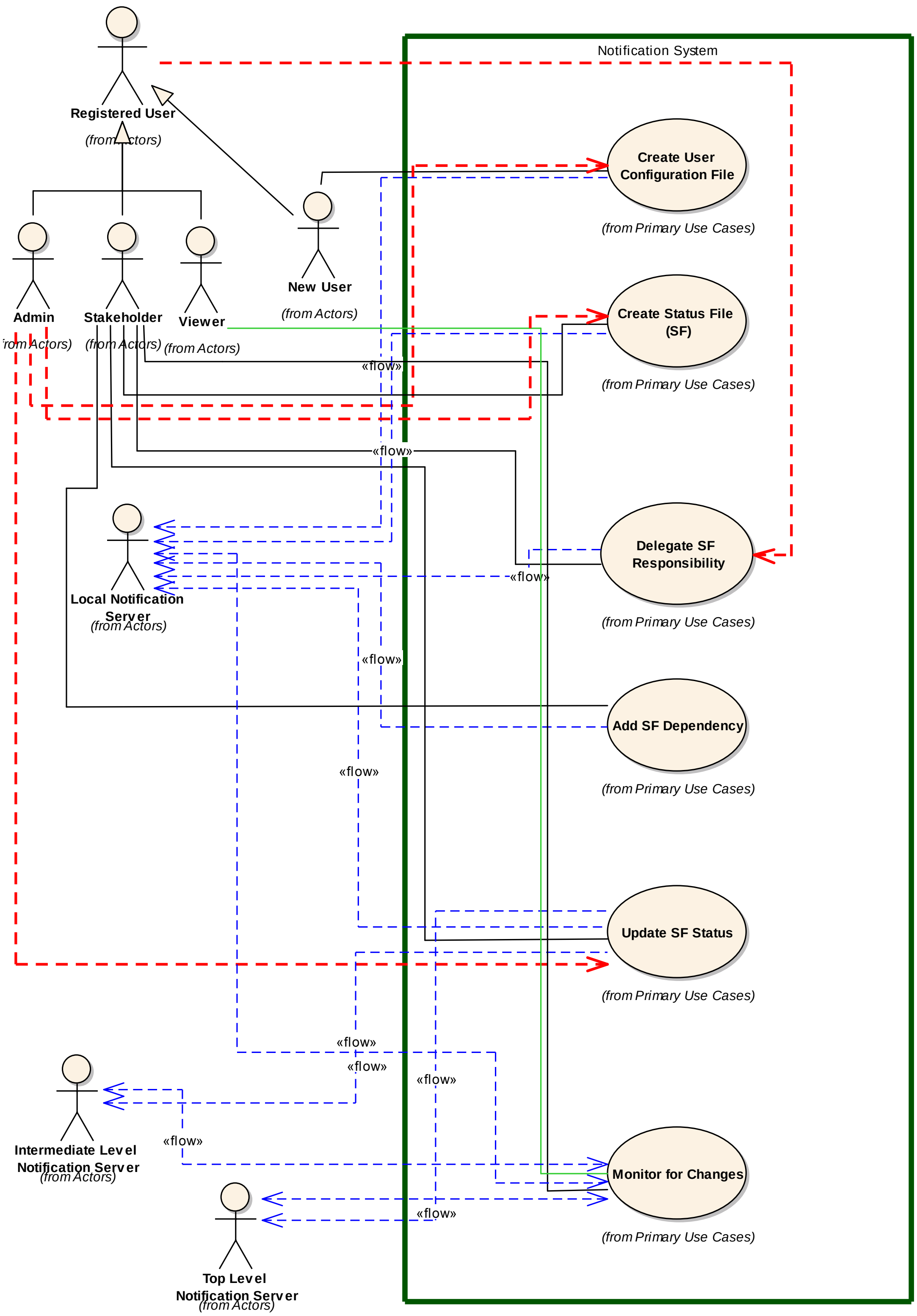


Figure 14. Overall Use Case Diagram

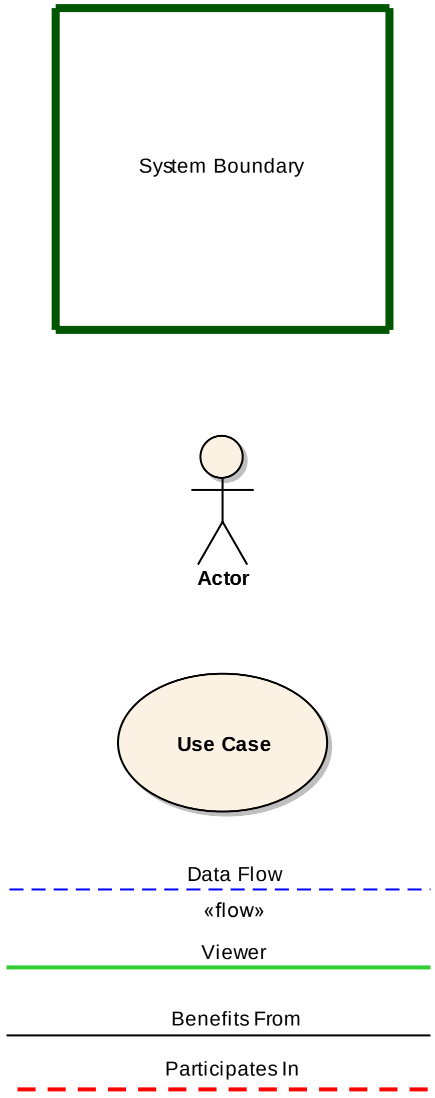


Figure 15. Overall Use Case Diagram Legend

The class diagram as shown in Figure 16 shows nominally the actions and attributes of the notification system. All of the actions and attributes are labeled as plainly as possible so as to reduce the level of confusion involved. These actions and attributes are laid out in such a way that a programmer familiar with UML and the principles of OOP could then start building around these actions and attributes to develop a working prototype.

The UserAgent class is nothing more than an interface into the UserConfiguration class. It has no attributes or actions of its own--it only acts as a means to interact with the UserConfiguration class and its extensions.

The UserConfiguration class is central to the entire notification system. It is what the UA accesses when it is started up, and it contains the primary components of receiving and providing notifications.

The admin class is an extension of the UserConfiguration class. They have access to all of the attributes and actions that other users have, but then extend the UserConfiguration class by obtaining additional attributes and actions related to being an LNSA.

The MissionAssetStakeholder class also extends the UserConfiguration class. It provides attributes and actions necessary to both monitor other missions and provide updates for the one for which authority has been granted. This is a zero to many relationship, in that a user may be a stakeholder for zero, one, or multiple missions and/or assets.

The MissionAssetViewer class is the last of the extensions of the UserConfiguration class. It provides the attributes and actions necessary to only monitor a mission or asset that they have been granted authorization to view. No provisions are made to update that mission or asset--they must report what they see to someone who does have that permission. For any particular mission or asset and individual can be either a stakeholder, a viewer, or have no permission to it, but not any combination thereof.

Both a stakeholder and a viewer perform their actions on a MissionAsset class for each mission or asset they have been granted permission for. Each MissionAsset class keeps a list of missions or assets on which it is dependent for maintaining mission capability. It is within the MissionAsset class that the checks are made on a periodic basis to determine if there is anything to report to the stakeholder(s) and/or viewer(s). A C2RMS-like monitoring agent is present on the drawing and could be used to automate checks of dependent assets.

5.5 Use Case 1: Create a Registered User

In this first use case, a new user becomes registered user on the notification architecture. The assumption in this use case is the operating notification architecture is running and a new individual has just arrived to the base. The system is mature.[7]

The new user visits the office responsible for normal network account creations. Along with giving this person network access, the personnel also create his or her account on the notification system, accessing the notification system through the admin's user agent (UA) and acting as a local notification system administrator (LNSA).

[7] It is acknowledged that in a situation where this architecture is brand new, both to the overall Air Force enterprise and to individual bases that the initial creation process will be different. The method of this implementation is intentionally omitted from this document.

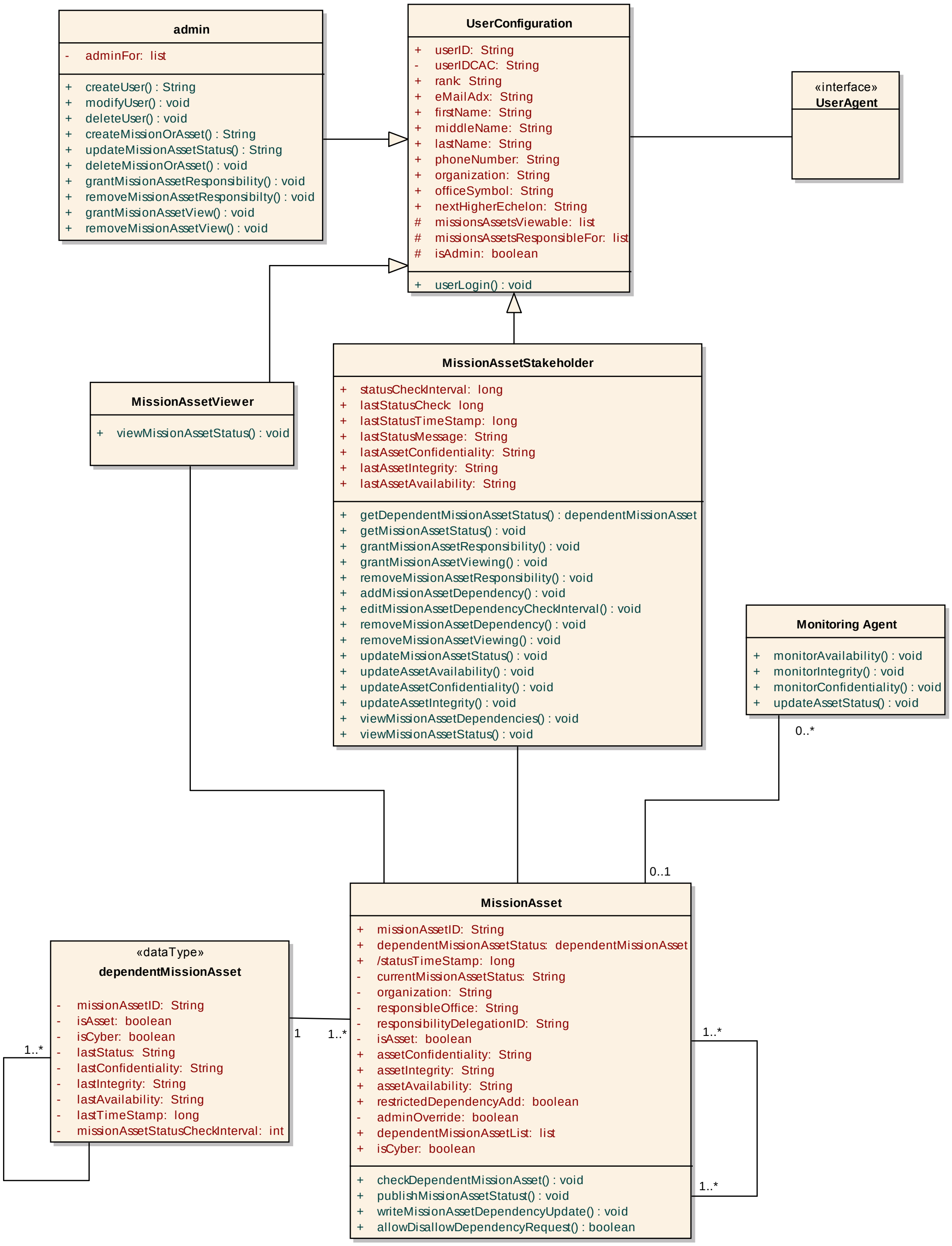

admin
- adminFor: list
+ createUser() : String
+ modifyUser() : void
+ deleteUser() : void
+ createMissionOrAsset() : String
+ updateMissionAssetStatus() : String
+ deleteMissionOrAsset() : void
+ grantMissionAssetResponsibility() : void
+ removeMissionAssetResponsibilty() : void
+ grantMissionAssetView() : void
+ removeMissionAssetView() : void
UserConfiguration
+ userID: String
- userIDCAC: String
+ rank: String
+ eMailAdx: String
+ firstName: String
+ middleName: String
+ lastName: String
+ phoneNumber: String
+ organization: String
+ officeSymbol: String
+ nextHigherEchelon: String
# missionsAssetsViewable: list
# missionsAssetsResponsibleFor: list
# isAdmin: boolean
+ userLogin() : void
«interface»
UserAgent
MissionAssetViewer
+ viewMissionAssetStatus() : void
MissionAssetStakeholder
+ statusCheckInterval: long
+ lastStatusCheck: long
+ lastStatusTimeStamp: long
+ lastStatusMessage: String
+ lastAssetConfidentiality: String
+ lastAssetIntegrity: String
+ lastAssetAvailability: String
+ getDependentMissionAssetStatus() : dependentMissionAsset
+ getMissionAssetStatus() : void
+ grantMissionAssetResponsibility() : void
+ grantMissionAssetViewing() : void
+ removeMissionAssetResponsibility() : void
+ addMissionAssetDependency() : void
+ editMissionAssetDependencyCheckInterval() : void
+ removeMissionAssetDependency() : void
+ removeMissionAssetViewing() : void
+ updateMissionAssetStatus() : void
+ updateAssetAvailability() : void
+ updateAssetConfidentiality() : void
+ updateAssetIntegrity() : void
+ viewMissionAssetDependencies() : void
+ viewMissionAssetStatus() : void
Monitoring Agent
+ monitorAvailability() : void
+ monitorIntegrity() : void
+ monitorConfidentiality() : void
+ updateAssetStatus() : void
0..*
0..1
MissionAsset
+ missionAssetID: String
+ dependentMissionAssetStatus: dependentMissionAsset
+ /statusTimeStamp: long
- currentMissionAssetStatus: String
- organization: String
- responsibleOffice: String
- responsibilityDelegationID: String
- isAsset: boolean
+ assetConfidentiality: String
+ assetIntegrity: String
+ assetAvailability: String
+ restrictedDependencyAdd: boolean
- adminOverride: boolean
+ dependentMissionAssetList: list
+ isCyber: boolean
+ checkDependentMissionAsset() : void
+ publishMissionAssetStatust() : void
+ writeMissionAssetDependencyUpdate() : void
+ allowDisallowDependencyRequest() : boolean
1..*
1..*
«dataType»
dependentMissionAsset
- missionAssetID: String
- isAsset: boolean
- isCyber: boolean
- lastStatus: String
- lastConfidentiality: String
- lastIntegrity: String
- lastAvailability: String
- lastTimeStamp: long
- missionAssetStatusCheckInterval: int
1
1..*
1..*

Figure 16. Overall Class Diagram

The LNSA, using the information provided for network access, creates and begins to populate the UCF. The UCF is populated with a list of those mission/asset identification numbers (MAINs) that the user will have the ability to monitor and/or update, eliminating the need for the user to be tied to a single computer to be able to perform monitoring and/or updating duties. Populating the UCF with MAINs can be done at the time of user registration if those MAINs are known and proper documentation has been provided showing that the individual has authority for those MAINs.

The UCF is tied to the user's CAC, eliminating the need for an additional password for this system and enforcing two-factor security to gain access to the UCF. Once populated, the UCF contains basic biographical information, fields identifying whether the user is an LNSA, what organization does the person belong to, who is the next person in the individual's chain of command, a list of missions the individual is a stakeholder for, a list of assets the individual is a stakeholder for, missions that can be viewed but cannot be updated, and assets that can be viewed but cannot be updated. There is also a user identification number (UID) that is provided to the user, is viewable in their user agent, and is used by others to add this newly registered user as a stakeholder or viewer of missions or assets.

Figure 17 illustrates the sequence diagram, showing interaction between the New User, Admin, and LNS. The new user provides the administrator with the necessary demographic information. The administrator enters this information through the administrator's user agent. The information is populated into the local notification server and a UCF is created. The LNS returns a UID to the administrator. At the conclusion of the sequence diagram the New User is now a Registered User and can use their CAC credentials to connect to the system through the user agent.

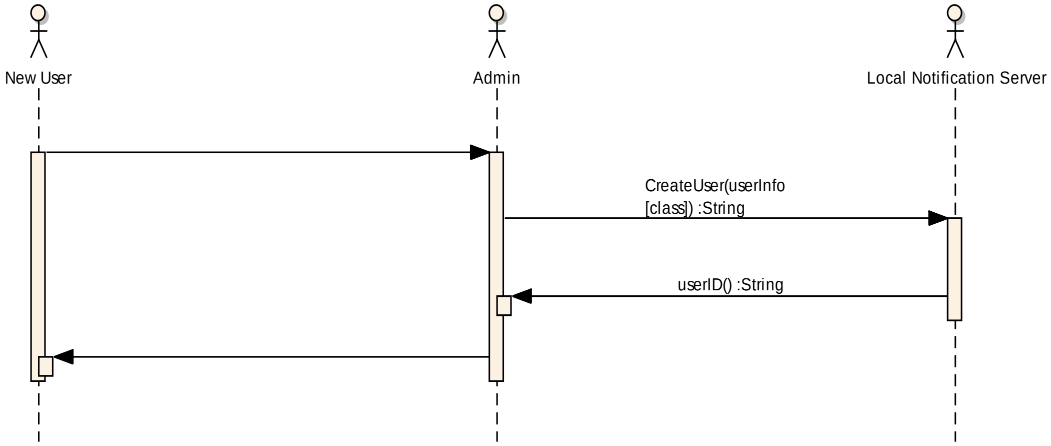


Figure 17. Add New User Sequence Diagram

Figure 18, the communication diagram, shows additional detail in the process. As noted above, it requires an administrator to be connected to the LNS through their UA to start the process. It then goes through the steps of entering the required fields, validating that the individual is being assigned to a valid organization, has a valid echelon user above them, and that they have not previously created a user account based on their CAC.

Figure 19 is the activity diagram for adding a new user. This drawing shows some of the logical flows of the process. If the organization or echelon to which the user has been submitted for do not exist, the system loops back to request the proper organizational information. If the user's CAC credentials do not exist in the notification system a UID is generated. If the credentials do exist, then an error is generated but the UID associated with the CAC is displayed.

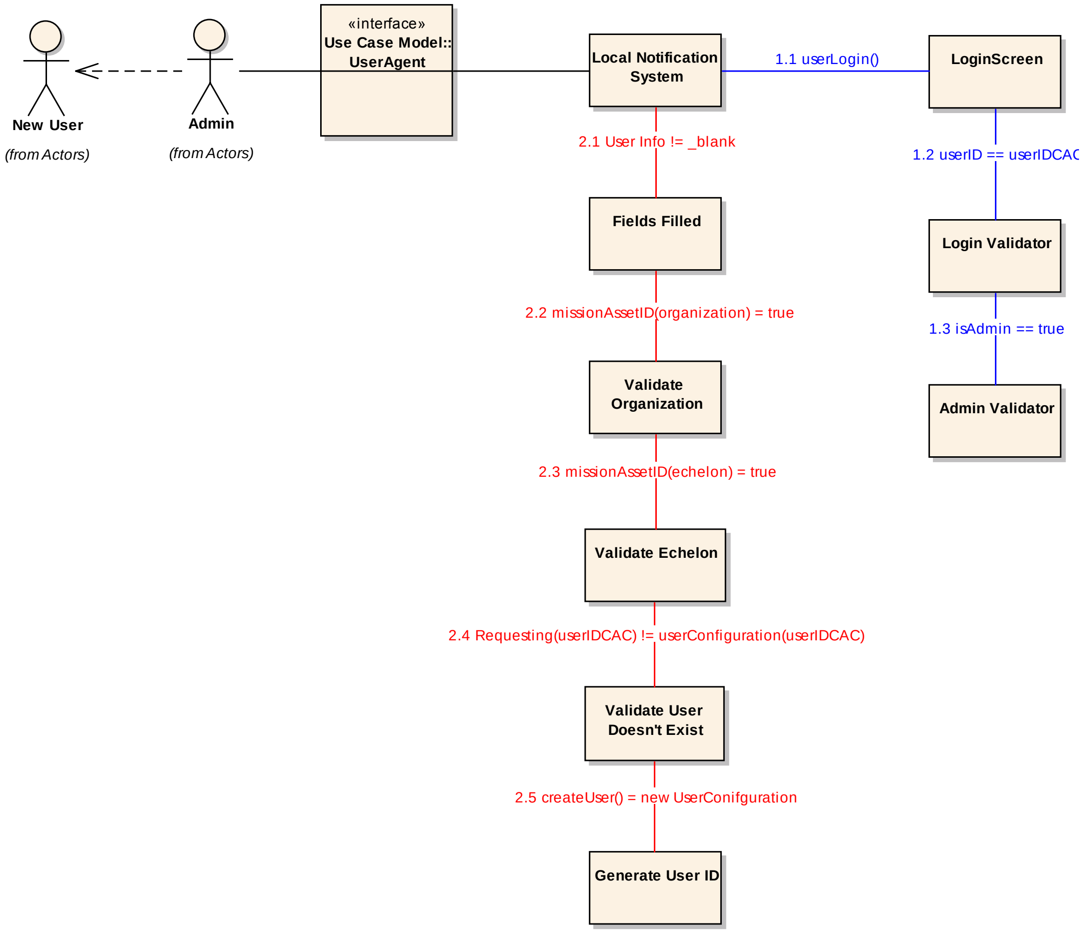


Figure 18. Add New User Communication Diagram

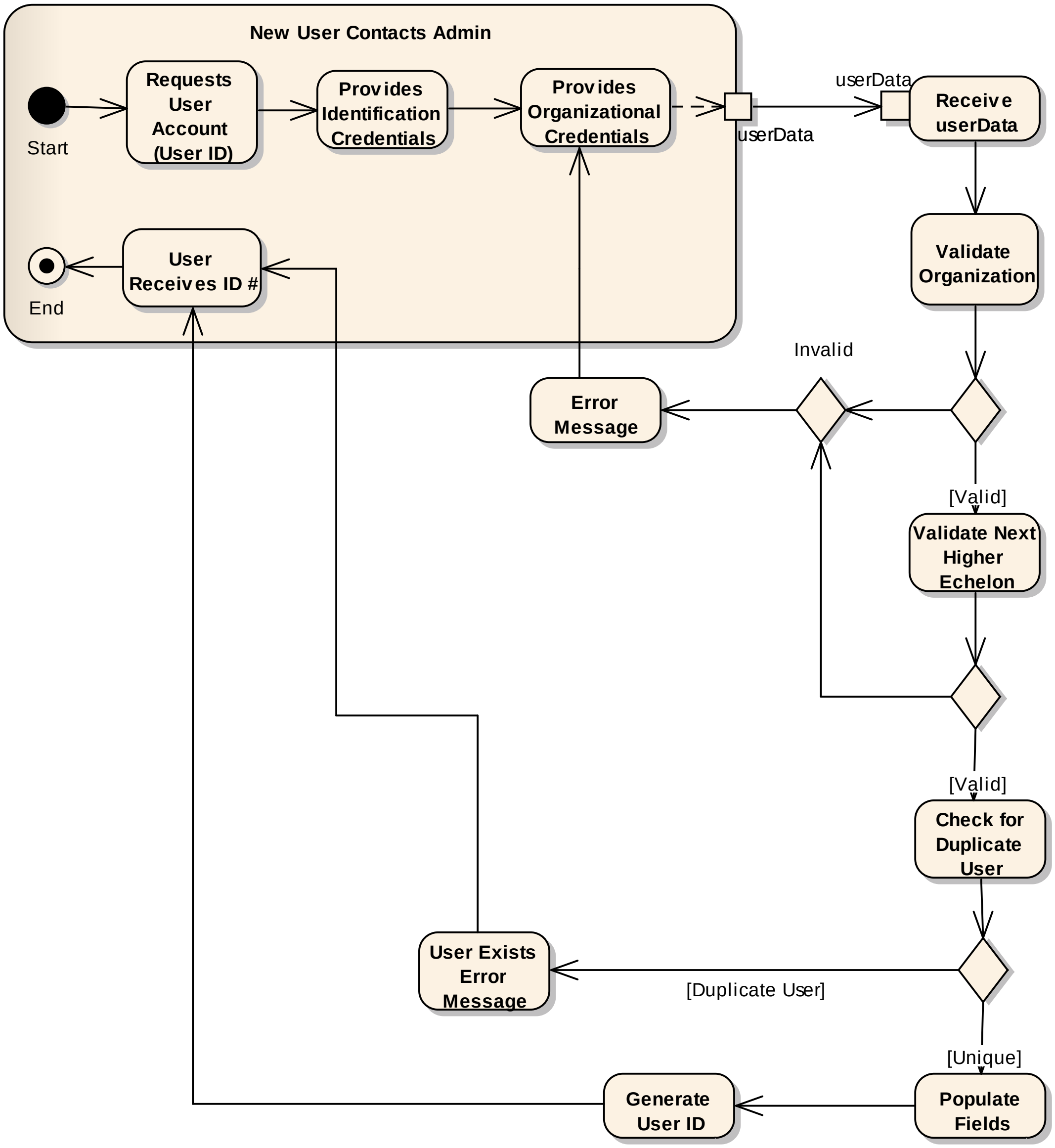


Figure 19. Add New User Activity Diagram

5.6 Use Case 2: New Mission Status File (MSF) or Asset Status File (ASF) is Created

No military environment is ever completely static. New missions and assets are added occasionally and when they are there needs to be a way to monitor them.

Authority for both missions and assets can be delegated from the commander to a subordinate through communication of the commander’s intent—that is what the commander wants the state of that asset or mission to be [3]. As previously discussed, this does not change who is responsible for it—that still lies with the commander.

The process of establishing a MSF or ASF begins with the communication of commander’s intent. Whether it is a mission or an asset for which authority is going to be delegated, the commander’s intent must be received by the individual who will be given the authority to act on behalf of the commander. This individual, prior to visiting with the local administrator of the notification system, needs to perform an initial assessment as to what services or assets will be needed to carry out the commander’s intent. This initial assessment is going to miss things that the asset or mission depends on and that is to be expected. As opined many times by many people, no plan ever survives first contact with reality. This individual should also know who it is in the reporting chain will have the next level of authority of the mission or asset. This may be hierarchical (i.e., the next level of supervision), it could be non-hierarchical yet still another individual in the organization, or it could be directly back to the commander.

With these pieces of information, the individual contacts the installation LNSA. The user provides proof of the commander’s intent, the known missions or assets that need to be monitored, and who needs to have echelon rights over the mission or asset. At this point, the LNSA will need to gather additional information that the requestor has probably not considered and/or has taken for granted. Many of these will be assets including, but not limited to, power, water, network connectivity, access to vehicles, and so on.

From this gathering of information, the LNSA can create a MSF or ASF as applicable to whether a mission or asset is being added, create the MAIN, and populate the missions and assets that need to be monitored. If an ASF is created for a cyber asset of any type (e.g., hardware, software, data source, etc.), this information is noted so that the information may be published in the next higher level of notification system so that AFNETOPS personnel may also monitor the status and so that if the asset is used by individuals at another base that they can also retrieve the status of that asset.

Figure 20 shows the interaction between the Registered User, Admin, and LNS in creating a mission or asset status file. The final step of that sequence is adding the mission as a dependency to the individual in the authority echelon (whether hierarchical or non-hierarchical) that has delegated the authority for the new item.

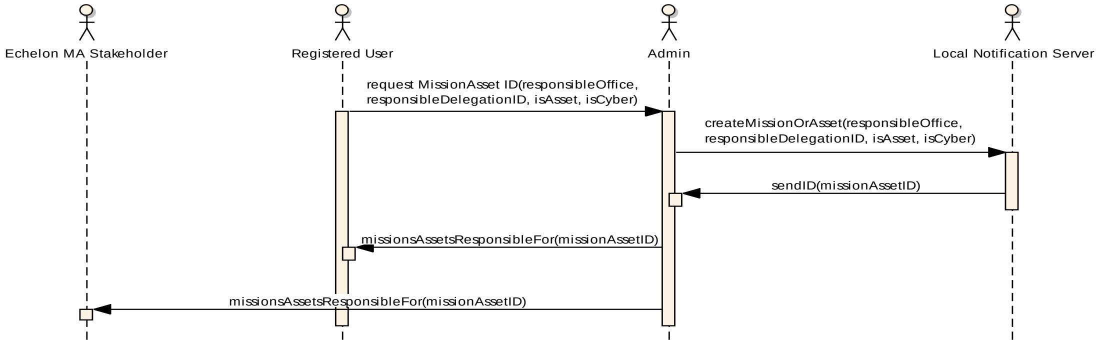
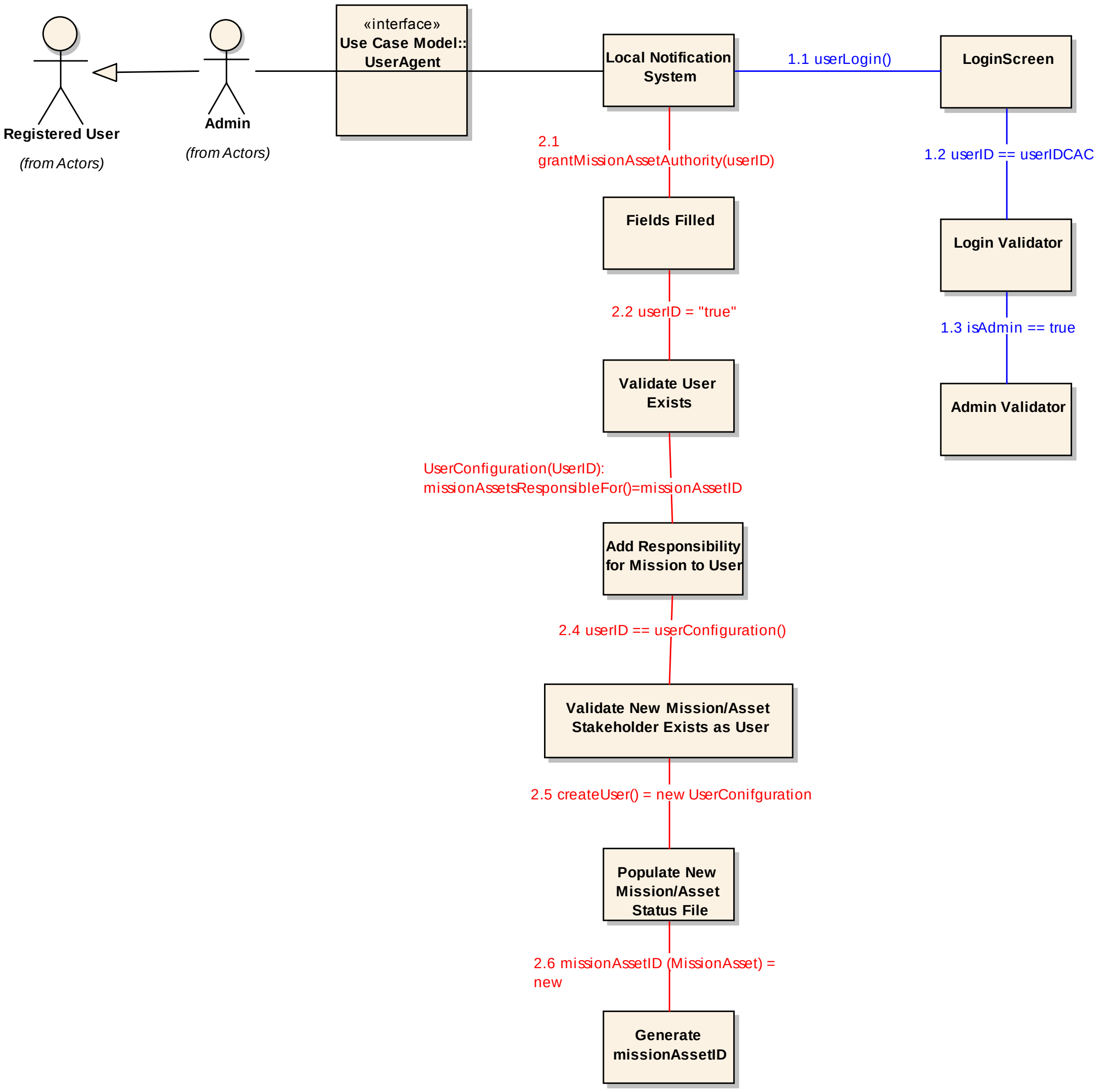

Echelon MA Stakeholder
Registered User
Admin
Local Notification Server
request MissionAsset ID(responsibleOffice, responsibleDelegationID, isAsset, isCyber)
createMissionOrAsset(responsibleOffice, responsibleDelegationID, isAsset, isCyber)
sendID(missionAssetID)
missionsAssetsResponsibleFor(missionAssetID)
missionsAssetsResponsibleFor(missionAssetID)

Figure 20. Create Status File Sequence Diagram

Figure 21 illustrates creating a status file in additional detail. Note that in this figure it is a registered user that is involved in the process, but the administrator is still in charge on information entry.



Registered User
(from Actors)
Admin
(from Actors)
«interface»
Use Case Model::
UserAgent
Local Notification System
1.1 userLogin()
LoginScreen
1.2 userID == userIDCAC
Login Validator
1.3 isAdmin == true
Admin Validator
2.1
grantMissionAssetAuthority(userID)
Fields Filled
2.2 userID = "true"
Validate User Exists
UserConfiguration(UserID):
missionAssetsResponsibleFor()=missionAssetID
Add Responsibility for Mission to User
2.4 userID == userConfiguration()
Validate New Mission/Asset Stakeholder Exists as User
2.5 createUser() = new UserConifguration
Populate New Mission/Asset Status File
2.6 missionAssetID (MissionAsset) =
new
Generate missionAssetID

Figure 21. Create Status File Communication Diagram

Figure 22 illustrates in additional detail the logical steps involved in creating a status file. Basic error checking is present at this step to ensure that there is organizational responsibility for the new MSF or ASF being created.

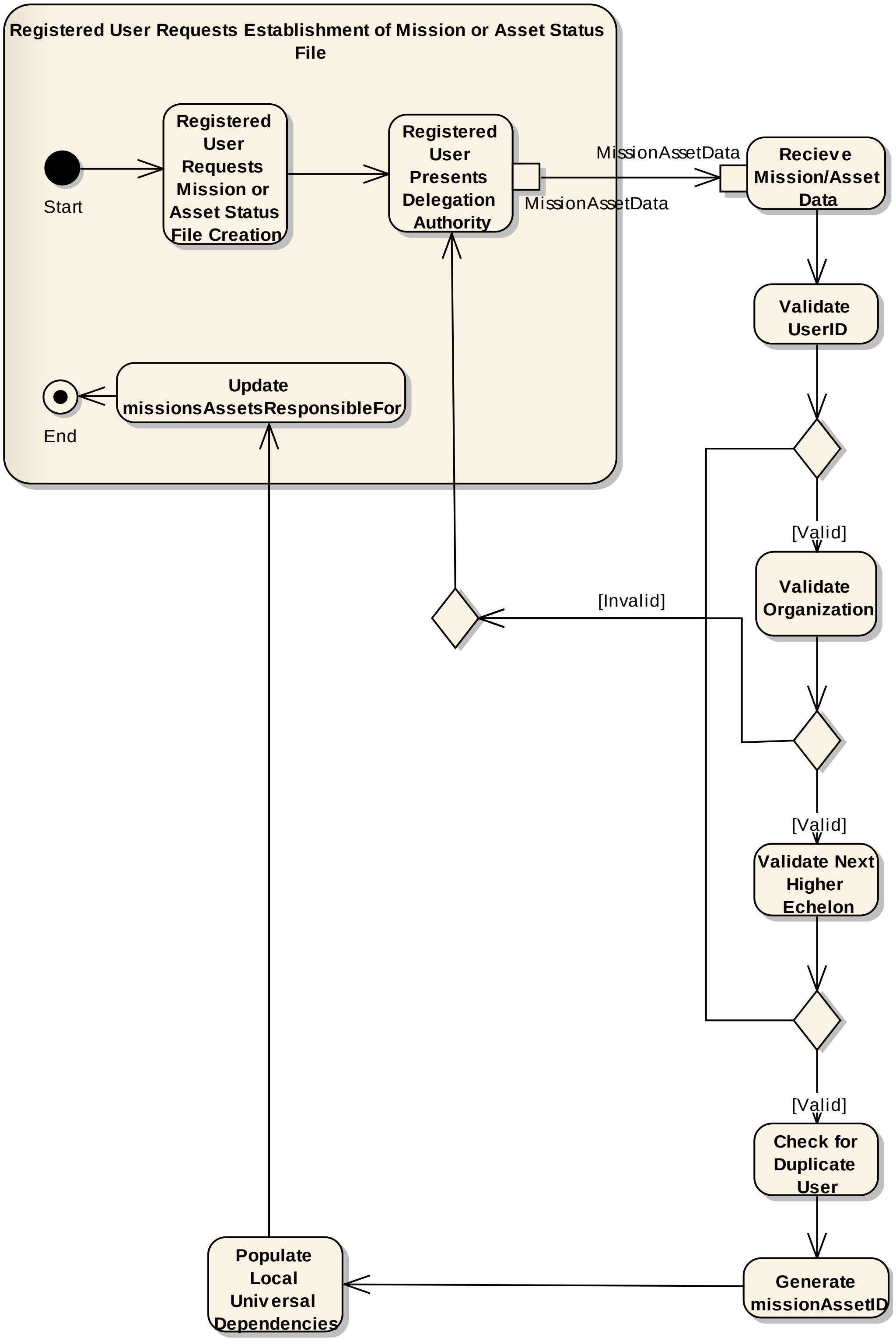

Registered User Requests Establishment of Mission or Asset Status File
Start
Registered User Requests Mission or Asset Status File Creation
Registered User Presents Delegation Authority
MissionAssetData
MissionAssetData
Recieve Mission/Asset Data
Validate UserID
[Valid]
Validate Organization
[Invalid]
[Valid]
Validate Next Higher Echelon
[Valid]
Check for Duplicate User
Generate missionAssetID
Populate Local Universal Dependencies
Update missionsAssetsResponsibleFor
End

Figure 22. Create Status File Activity Diagram

5.7 Use Case 3: Authority for the Mission or Asset is Delegated or Shared

A case was made during the investigative questions for a system that was capable of sharing visibility and authority for a mission or an asset between multiple individuals. A case was also made for multiple people having access to notifications of changes in the status of a monitored mission or asset as no one person can be vigilant over a mission 24 hours a day.

When a MSF or ASF is created, it is designed for one individual who has received commander's intent to have control of that file. The reality is that in most cases multiple persons need to have access to the status file. And as people move into new positions and/or are promoted to a rank in which authority for the mission or asset can be shared, these individuals need to acquire the ability to monitor the status files and make changes to the status as appropriate.

This process begins with there being a need to share the authority to view and update a MSF or ASF. Once this need is identified, two types of users may further share this authority: either a stakeholder for that mission or asset or an echelon user of that mission or asset. The user with authority to share the authority for the mission or asset requests the user identification number from the person who will receive authority for the mission or asset. The delegator/sharer of the authority for the mission or asset goes into their UA, executes grantMissionAssetAuthority() and inputs both the MAIN of the mission or asset to be shared and the UID of the individual who is to receive the authority.

A message is sent to the authority gaining user stating that the delegator/sharer of the authority has identified gaining user to receive the authority. The gaining user has an opportunity to approve or disapprove adding this authority to their UA. If approved, this authority is added to the user's UCF and is present when they are using their UA.

From this point until the authority is removed, each time the user is running the UA, the missions and assets for which they have authority are actively monitored for changes to status.

Figure 23 shows a mission or asset stakeholder contacting the LNS to share the authority for the mission.

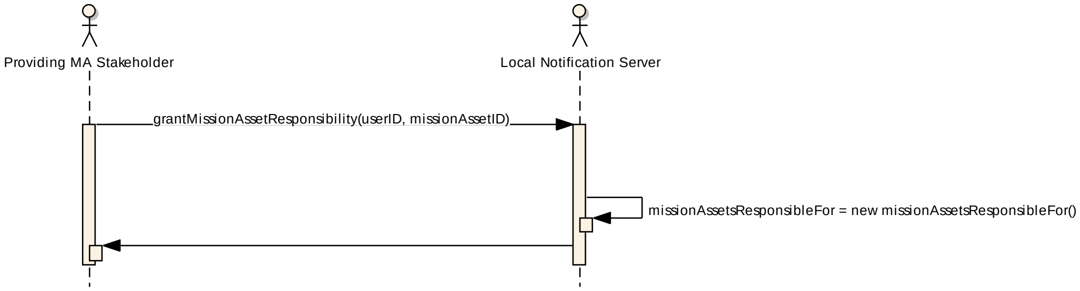

Providing MA Stakeholder
Local Notification Server
grantMissionAssetResponsibility(userID, missionAssetID)
missionAssetsResponsibleFor = new missionAssetsResponsibleFor()

Figure 23. Mission Authority Delegation Sequence Diagram

Figures 24 and 25 show communications and activity diagrams respectively that illustrate in additional detail the logical steps involved. These are far less complicated than in the previously detailed use cases because less error checking and validation needs to occur.

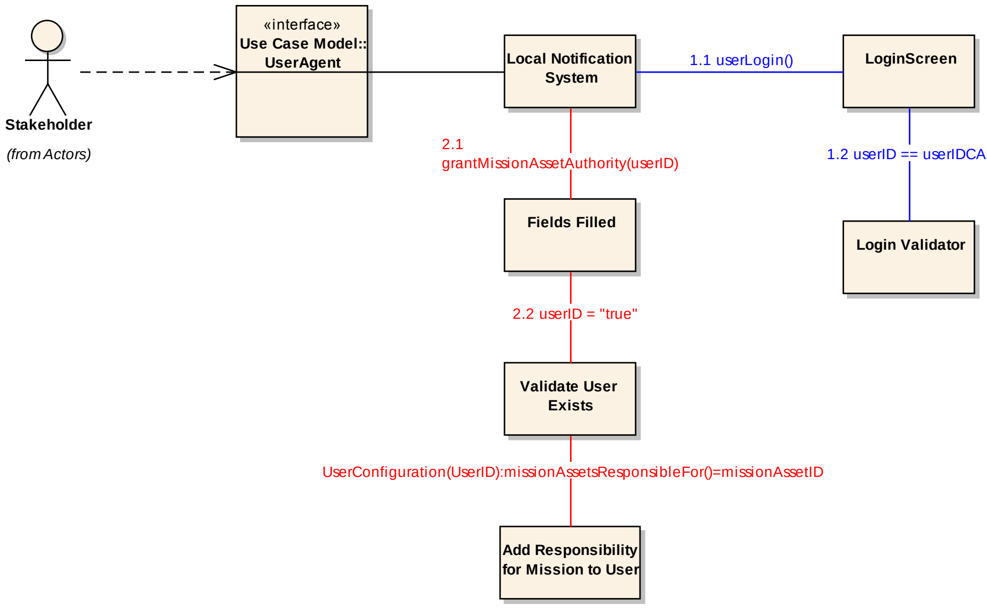


Figure 24. Mission Authority Delegation Communication Diagram

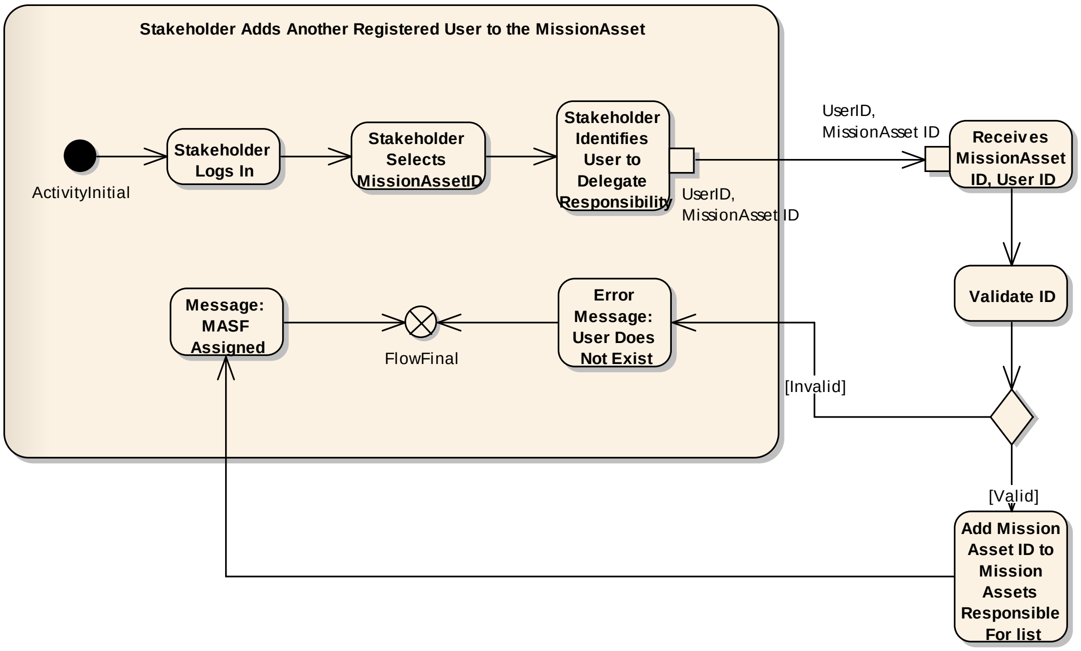

Stakeholder Adds Another Registered User to the MissionAsset
ActivityInitial
Stakeholder Logs In
Stakeholder Selects MissionAssetID
Stakeholder Identifies User to Delegate Responsibility
UserID, MissionAsset ID
UserID, MissionAsset ID
Receives MissionAsset ID, User ID
Validate ID
[Invalid]
[Valid]
Add Mission Asset ID to Mission Assets Responsible For list
Error Message: User Does Not Exist
FlowFinal
Message: MASF Assigned

Figure 25. Mission Authority Delegation Activity Diagram

5.8. Use Case 4: Add a Mission or Asset Dependency

As mentioned earlier, both missions and assets rely on other missions and assets to function. The failure or non-availability of a mission or asset caused by the failure or non-availability of a lone dependency will only occur if that dependency is deemed a single point of failure. Otherwise, the health of a particular mission or asset can be only determined by looking surmising the Level 1 EMSA for all of the components that go into the mission and determine the Level 2 EMSA at that given time [35].

These dependencies, as defined earlier, change over time as there are changes to standard operating procedures, instructions, regulations, and availability. This use case will cover only the process of adding a mission or asset dependency. While removing one is reasonably trivial in comparison to adding one, it is a task that is less likely to happen and was not included in the six use cases for this research.

Within the structure of MSF's and ASF's is a list of dependencies. These can be other missions, other assets, or likely a combination of both. The overall class diagram's data type dependentMissionAsset is a data structure containing a series of elements key to determining Level 1 EMSA. They are the MAIN, whether the item is an asset or a mission, whether the item is a cyber asset or not, the last known status, the last known values for the CIA triad (as applicable to what the dependency is), the last time it was checked, and the length of time between checks. These values represent the bare minimum types of information necessary to obtain rudimentary Level 1 EMSA for a mission or an asset. In short, what is it, what is its status, has it been compromised, when was it checked last, and how often is it checked.

When through the course of doing business or prompted by a helper application (such as Milcord's CLearn [49]) it is determined that there is a dependency for a mission or asset that was not previously identified, a mission stakeholder takes steps to add that dependency. The first step is to find the MAIN of the dependency to be added.

How the MAIN is advertised or acquired depends completely on the type of mission or asset and how the owner of that mission or asset has chosen to control it. For web-based systems, the MAIN may be present on the welcome or login screen. Or, as part of the registration process, the user of the system receives the MAIN in the confirmation e-mail from the system administrator. For physical assets, determining the MAIN may require an e-mail or phone call to the controlling organization. OPSEC considerations will play into the equation as to how widely to disperse this information and through what channels.

When the MAIN is found, the stakeholder engages the UA, selects the mission or asset that has a new identified dependency, and requests adding the dependency with the MAIN. A key point to stress here is that the requestor is not requesting adding the dependency for themselves—they are requesting to add the dependency for the mission or asset that they are stakeholder for. Once or if added to the list of dependencies, all who are stakeholders for that particular mission or asset will be able to see the status updates for that dependency. This achieves one of the goals that all mission or asset stakeholders have a shared representation of their dependencies regardless of who is in charge when the alert arrives.

Once the LNS confirms that the MAIN exists in the system, it asks the stakeholder how often the dependency should be checked. This time interval for requesting status should be based on how important the dependency is to the mission or asset. There is a default minimum time interval that can be selected so as to not induce a self-generated DOS attack.

When the MAIN is confirmed and the time interval for checking is entered, the LNS checks to see whether access to this mission status is unrestricted or restricted. If the status of this dependency has been identified as unrestricted, the LNS adds the dependency to the requested status file and sends a message to stakeholders for that dependency that a mission or asset is monitoring their mission or asset as a dependency.

If instead the status of the dependency has been identified as restricted, the LNS sends a notification to the stakeholder of the MAIN in question and notifies the requestor that approval must be provided by the MAIN stakeholder. The MAIN stakeholder's notification consists of the requestor's name, organization, base, and contact information. The MAIN stakeholder has the option of either approving or disapproving the request based on criteria that is completely up to the stakeholder. If the stakeholder has questions as to why the requestor needs access to the mission status, the stakeholder can contact the requestor. If approved, then the process of adding to the appropriate lists mirrors that as if the MAIN was unrestricted. Figure 26 below shows the logical flow of adding a dependency.

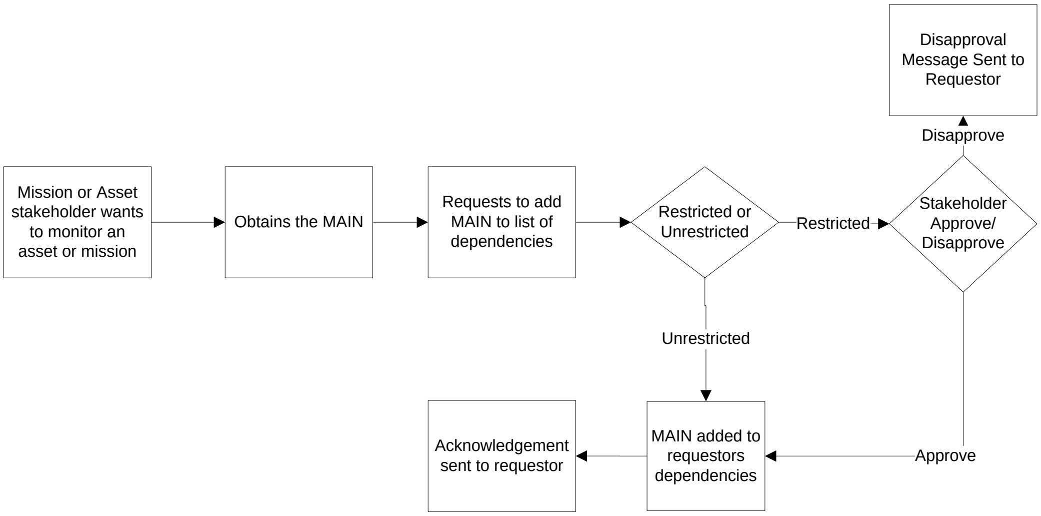

Mission or Asset stakeholder wants to monitor an asset or mission
Obtains the MAIN
Requests to add MAIN to list of dependencies
Restricted or Unrestricted
Restricted
Stakeholder Approve/ Disapprove
Disapprove
Disapproval Message Sent to Requestor
Unrestricted
MAIN added to requestors dependencies
Approve
Acknowledgement sent to requestor

Figure 26. Logic Flow – Adding a Dependency

Figure 27 repesents the sequence diagram for adding a mission dependency. Note that in this sequence diagram there is a consider section to see if the item is a restricted or unrestricted dependency.

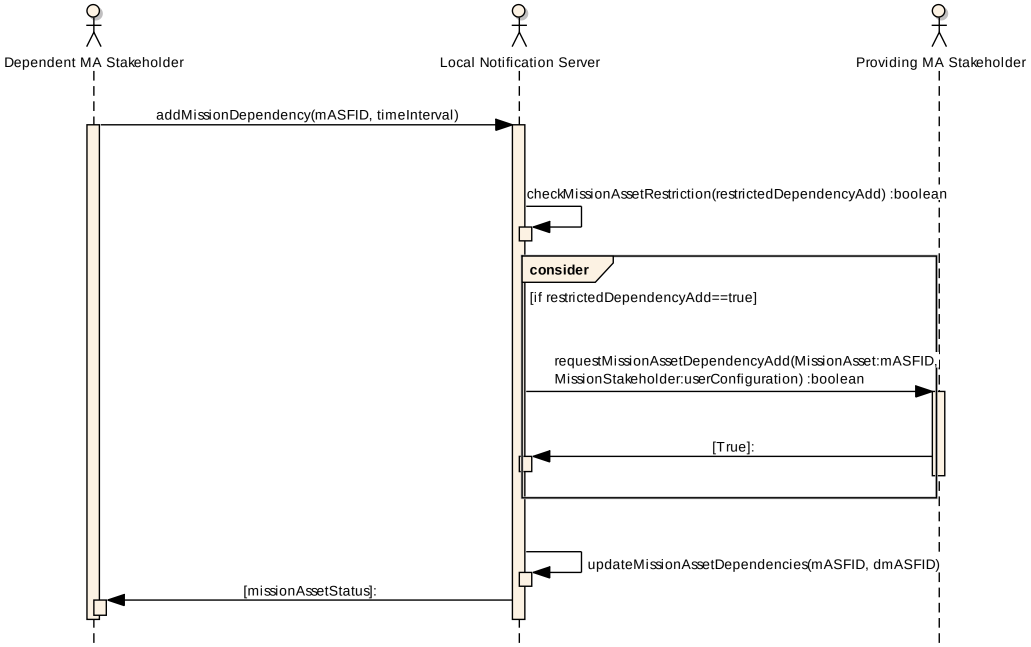
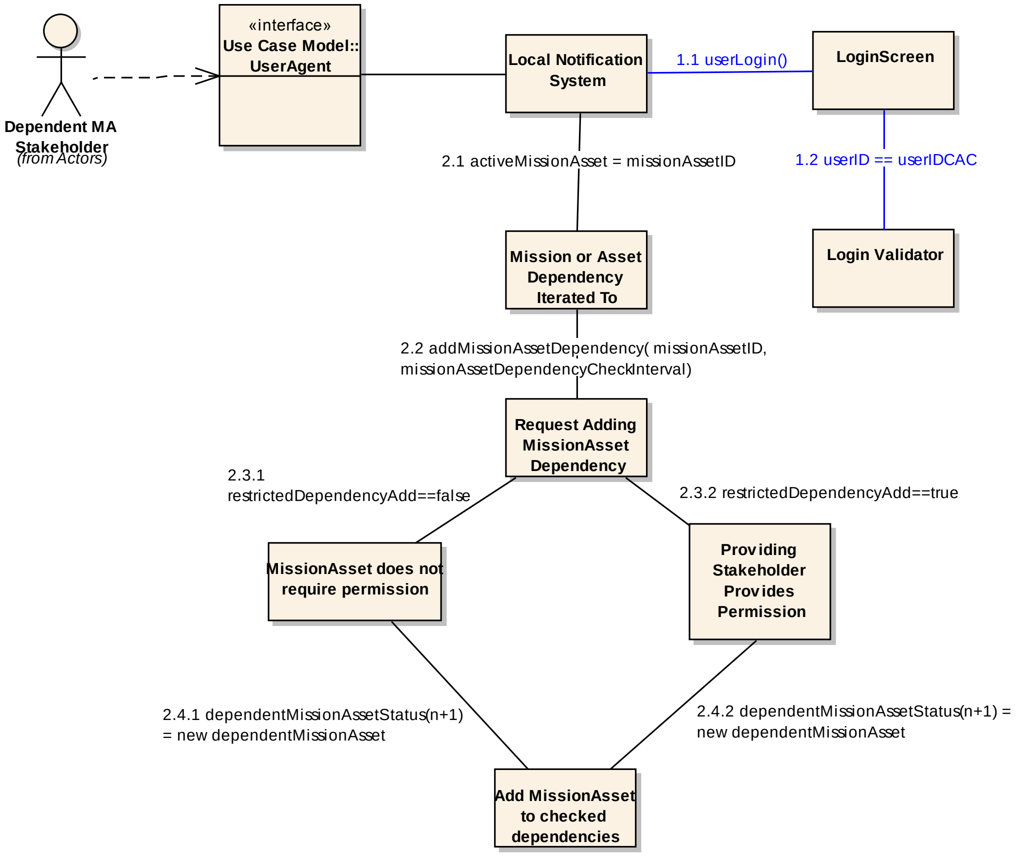

Dependent MA Stakeholder
Local Notification Server
Providing MA Stakeholder
addMissionDependency(mASFID, timeInterval)
checkMissionAssetRestriction(restrictedDependencyAdd) :boolean
consider
[if restrictedDependencyAdd==true]
requestMissionAssetDependencyAdd(MissionAsset:mASFID, MissionStakeholder:userConfiguration) :boolean
[True]:
updateMissionAssetDependencies(mASFID, dmASFID)
[missionAssetStatus]:

Figure 27. Sequence Diagram for Adding a Mission or Asset Dependency

Figure 28 is the communications diagram for adding the dependency. This communication diagram is more complicated than previous diagrams because it has to take into account both restricted and unrestricted ASFs and MSFs which could finish in the same positive outcome.



Dependent MA Stakeholder
(from Actors)
«interface»
Use Case Model::
UserAgent
Local Notification System
1.1 userLogin()
LoginScreen
2.1 activeMissionAsset = missionAssetID
1.2 userID == userIDCAC
Mission or Asset Dependency Iterated To
Login Validator
2.2 addMissionAssetDependency(missionAssetID, missionAssetDependencyCheckInterval)
Request Adding MissionAsset Dependency
2.3.1 restrictedDependencyAdd==false
2.3.2 restrictedDependencyAdd==true
MissionAsset does not require permission
Providing Stakeholder Provides Permission
2.4.1 dependentMissionAssetStatus(n+1) = new dependentMissionAsset
2.4.2 dependentMissionAssetStatus(n+1) = new dependentMissionAsset
Add MissionAsset to checked dependencies

Figure 28. Communication Diagram for Adding a Mission or Asset Dependency

Figure 29 is the activity diagram for this use case, and it too is more complicated than previous activity diagrams. This drawing introduces the concept of splitting out the flow into multiple directions simultaneously.

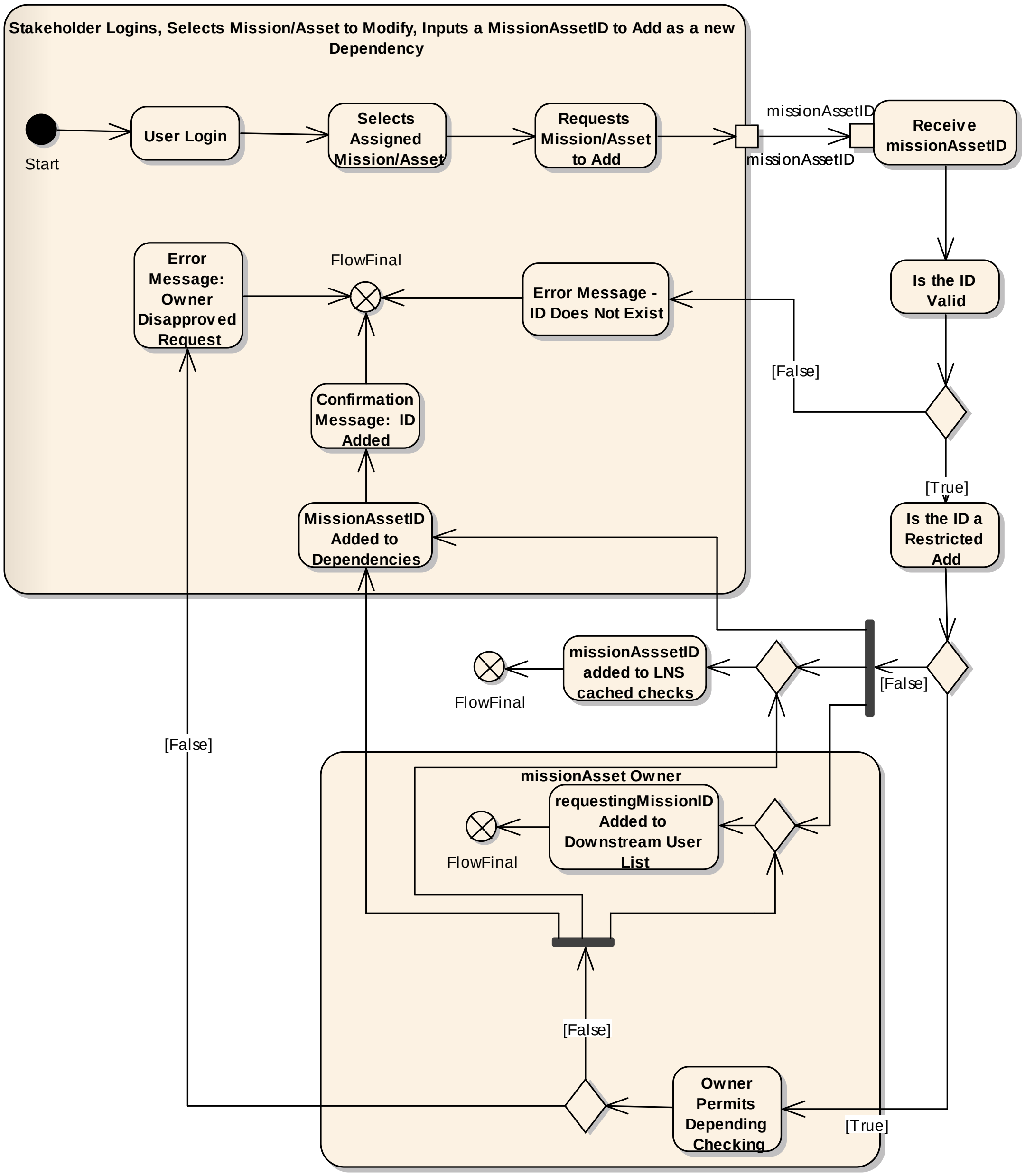

Stakeholder Logins, Selects Mission/Asset to Modify, Inputs a MissionAssetID to Add as a new Dependency
Start
User Login
Selects Assigned Mission/Asset
Requests Mission/Asset to Add
missionAssetID
missionAssetID
Receive missionAssetID
Error Message: Owner Disapproved Request
FlowFinal
Error Message - ID Does Not Exist
Is the ID Valid
[False]
Confirmation Message: ID Added
[True]
MissionAssetID Added to Dependencies
Is the ID a Restricted Add
missionAsssetID added to LNS cached checks
FlowFinal
[False]
[False]
missionAsset Owner
requestingMissionID Added to Downstream User List
FlowFinal
[False]
Owner Permits Depending Checking
[True]

Figure 29. Activity Diagram for Adding a Mission or Asset Dependency

5.9. Use Case 5: Update an MSF or ASF Status

Events occur that will change the availability of a mission or asset or the integrity or confidentiality of a cyber asset. When this occurs, whether through something that is seen directly in the span of control of the stakeholder, or is something that is received as a notification through the UA as an incident of something the stakeholder is monitoring, the stakeholder should attempt to take that Level 1 EMSA and expand it to a Level 2 EMSA [35]. Or put in more simple terms, the mission or asset stakeholder has to decide how or if the incident affects their mission or asset.

In this initial iteration of the proposed architecture, this elevation from Level 1 to Level 2 EMSA is decision that must be made completely at the discretion of the stakeholder based on his or her understanding of the mission or asset in question. There are ways that have been presented previously in which automation could help make this decision. Chapter III discussed a rules-based system as used in the Air Defense community in which mission capability is a measure of having or not having certain assets and missions available. Case-based reasoning is also an area where decision making can be improved by looking at past events and attempting to apply them to current events. Undoubtedly there are other ways in which the decision making process can be improved.

For this architecture, it is the stakeholder, one who has the authority for a mission or asset and has received commander's intent on this task or item, who has to do this based on knowledge and experience. If the stakeholder makes the decision that the mission or asset that they are responsible for is no longer fully capable of meeting the requirements as communicated through commander's intent, it is their responsibility in this architecture to change the status in the MSF or ASF as applicable.

When a situation like that occurs, the stakeholder engages their UA, selects the appropriate mission or asset that they are responsible for, and uses updateMissionAssetStatus(). For MSF's, calling this function will update the variables currentStatus() and availability() in the LNS. For ASF's, currentStatus(), availability(), integrity(), and confidentiality() are updated. In either case, the timestamp of the update is calculated based on the system time of the LNS. This change in status acts as the publish action in the publish/subscribe/pull model.

At the same time, the LNS logs the change of the status and the other appropriate items that associated with the change of status. The exact nature of what is logged is left as an implementation detail, though certain things that come to mind to ensure non-repudiation and authentication would include IP address, stakeholder bio information, and so on. Whatever items are determined to be necessary must also be balanced with the space that they will take up and the available quantity of disk space for such a logging scheme.

If the update is being made to an ASF that is a cyber asset, the LNS sees this when the update is made and publishes the update to the next higher notification system. If the next level is an ILNS, then this cycle of pushing to the next level continues until it hits a TLNS. Figures 30 and 31 show flow diagrams to reflect this process in additional depth.

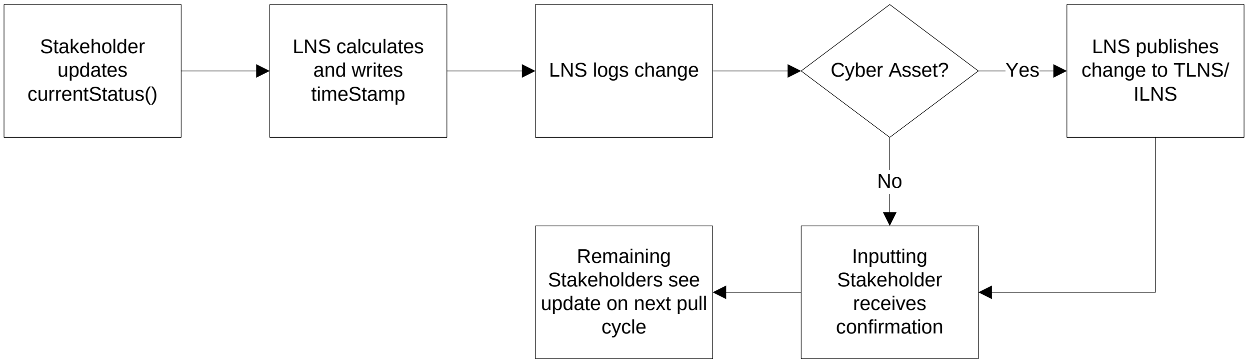

Stakeholder updates currentStatus()
LNS calculates and writes timeStamp
LNS logs change
Cyber Asset?
Yes
LNS publishes change to TLNS/ ILNS
No
Inputting Stakeholder receives confirmation
Remaining Stakeholders see update on next pull cycle

Figure 30. Flow diagram - Status Update

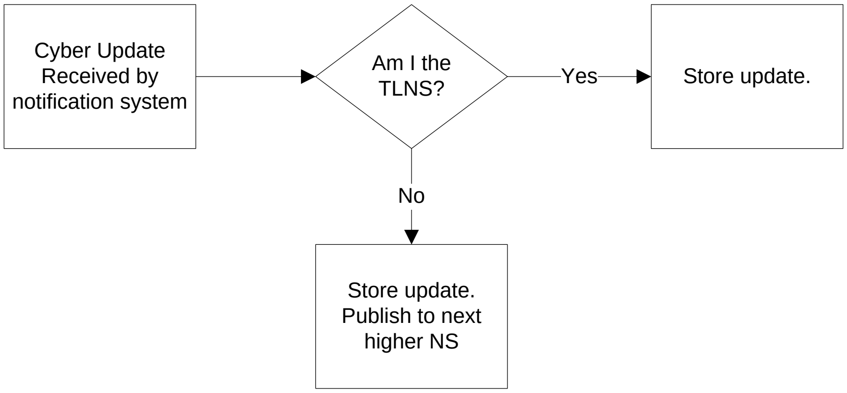

Cyber Update Received by notification system
Am I the TLNS?
Yes
Store update.
No
Store update. Publish to next higher NS

Figure 31. Flow Diagram - Updating to the TLNS

If a cyber asset is involved and if instead of the stakeholder it was an administrator at the LNS, ILNS, or TLNS that noticed the incident, a similar but decidedly different step of events occurs. The administrator at the level that noticed the problem has the proper rights to update the ASF to reflect that there has been an incident detected, the basic nature of the incident, and any other pertinent information thought to be of benefit to a downstream using asset or mission.

At the same time, the administrator can access the AdminOverride field to denote that an admin has overridden the value if the situation dictates. With this field set, only an admin may update the status until an admin turns off the flag. This is done to transfer control to the administrators so that with their elevated awareness they can provide the most accurate picture of what is going on to downstream stakeholders who are dependent on this particular asset. Overriding the asset stakeholder's control over the status is not required and should not be used indiscriminately. But at the same time, it provides a safety valve to enforce integrity into the process.

Figure 32 shows the sequence diagram for updating a status file. Notice again the consider section. In this case, the consider section checks to see if the item is a cyber asset. If so, then communication is extended to the TLNS for storage and usage as described in previous chapters. Figures 33 and 34 show communications and activity diagrams for updating status files. In particular, Figure 34 shows the Admin Override logic that puts administrators in charge of further updates when they detect a cyber incident and chose to invoke that level of control.

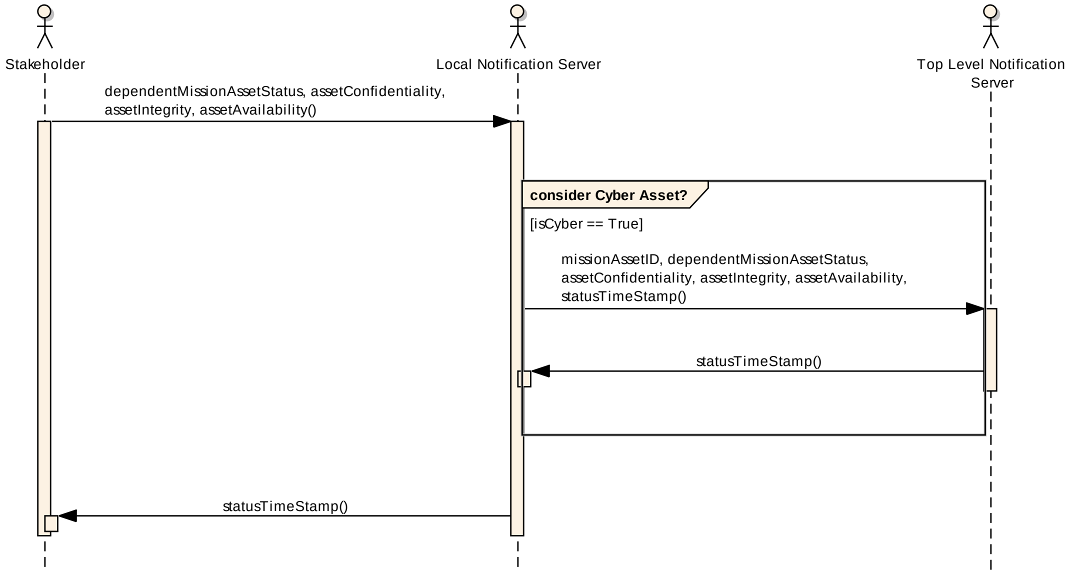

Stakeholder
Local Notification Server
Top Level Notification Server
dependentMissionAssetStatus, assetConfidentiality, assetIntegrity, assetAvailability()
consider Cyber Asset?
[isCyber == True]
missionAssetID, dependentMissionAssetStatus, assetConfidentiality, assetIntegrity, assetAvailability, statusTimeStamp()
statusTimeStamp()
statusTimeStamp()

Figure 32. Sequence Diagram - Update Status File

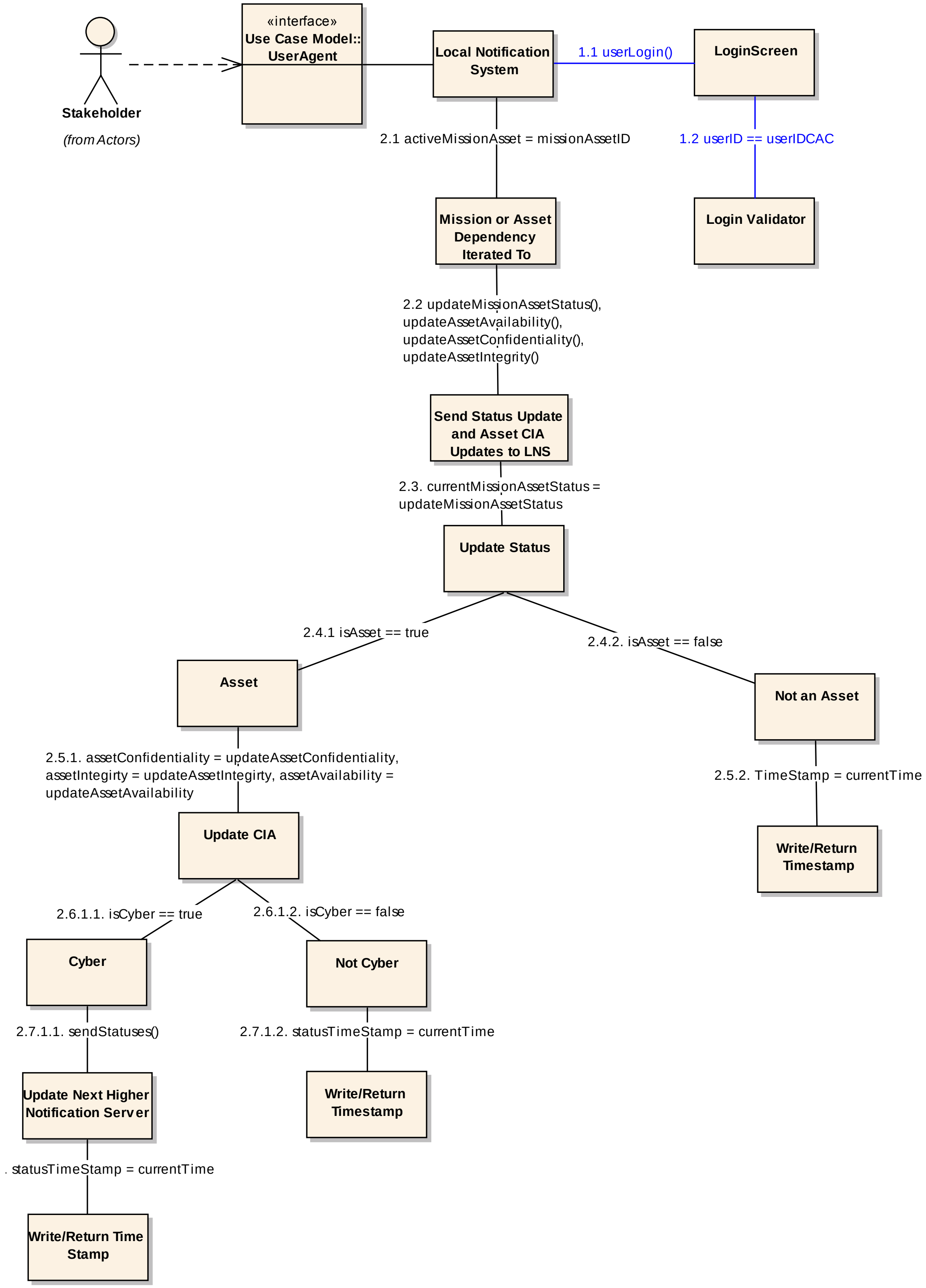

Stakeholder
(from Actors)
«interface»
Use Case Model::
UserAgent
Local Notification System
1.1 userLogin()
LoginScreen
2.1 activeMissionAsset = missionAssetID
1.2 userID == userIDCAC
Mission or Asset Dependency Iterated To
Login Validator
2.2 updateMissionAssetStatus(),
updateAssetAvailability(),
updateAssetConfidentiality(),
updateAssetIntegrity()
Send Status Update and Asset CIA Updates to LNS
2.3. currentMissionAssetStatus =
updateMissionAssetStatus
Update Status
2.4.1 isAsset == true
2.4.2. isAsset == false
Asset
Not an Asset
2.5.1. assetConfidentiality = updateAssetConfidentiality,
assetIntegirty = updateAssetIntegirty, assetAvailability =
updateAssetAvailability
2.5.2. TimeStamp = currentTime
Update CIA
Write/Return Timestamp
2.6.1.1. isCyber == true
2.6.1.2. isCyber == false
Cyber
Not Cyber
2.7.1.1. sendStatuses()
2.7.1.2. statusTimeStamp = currentTime
Update Next Higher Notification Server
Write/Return Timestamp
. statusTimeStamp = currentTime
Write/Return Time Stamp

Figure 33. Communication Diagram - Update Status File

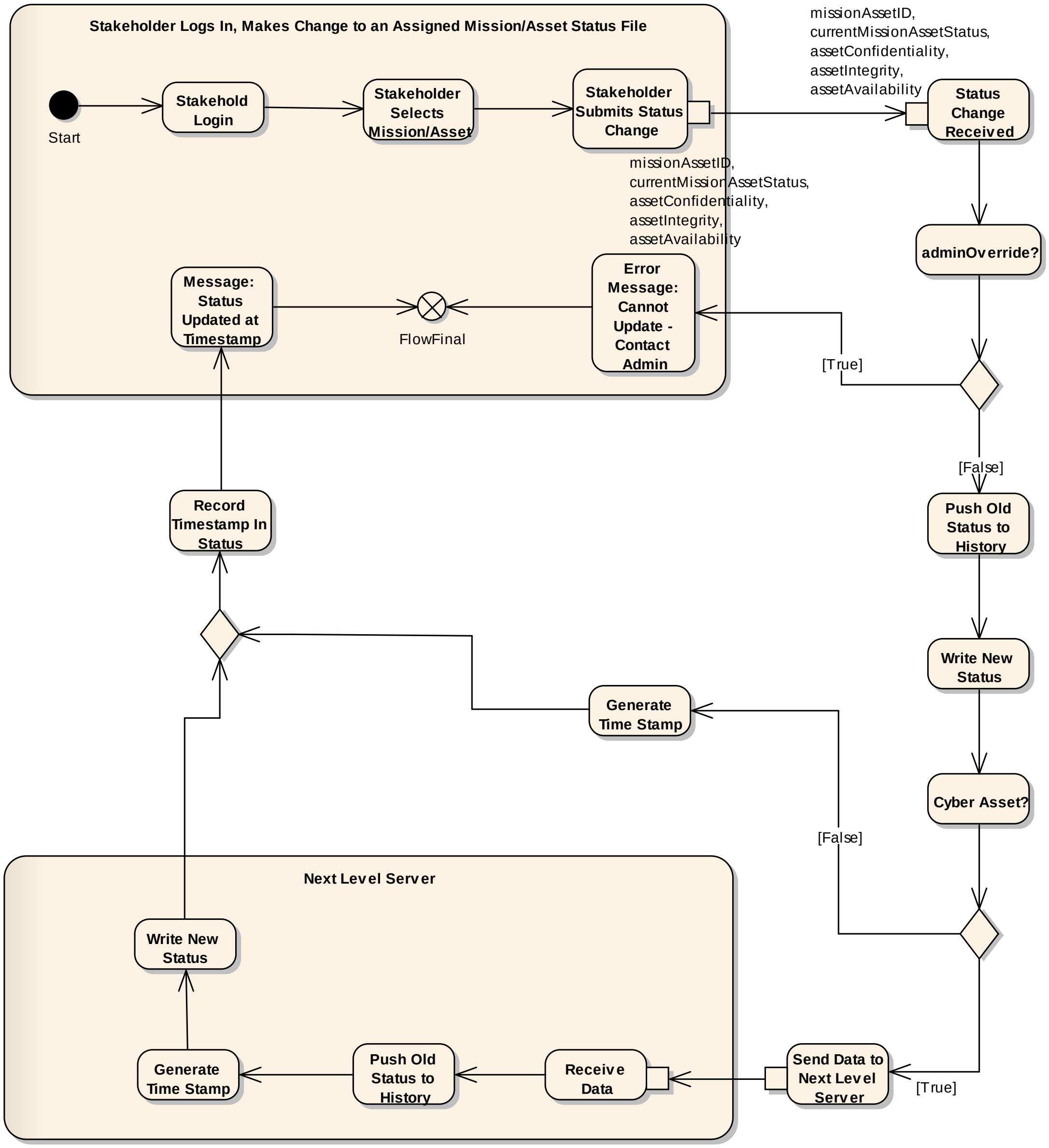

Stakeholder Logs In, Makes Change to an Assigned Mission/Asset Status File
Start
Stakehold Login
Stakeholder Selects Mission/Asset
Stakeholder Submits Status Change
missionAssetID,
currentMissionAssetStatus,
assetConfidentiality,
assetIntegrity,
assetAvailability
Status Change Received
missionAssetID,
currentMissionAssetStatus,
assetConfidentiality,
assetIntegrity,
assetAvailability
adminOverride?
Error Message: Cannot Update - Contact Admin
Message: Status Updated at Timestamp
FlowFinal
[True]
[False]
Push Old Status to History
Record Timestamp In Status
Write New Status
Generate Time Stamp
Cyber Asset?
[False]
Next Level Server
Write New Status
Generate Time Stamp
Push Old Status to History
Receive Data
Send Data to Next Level Server
[True]

Figure 34. Activity Diagram - Update Status File

5.10 Use Case 6: Monitored Status File Receives an Alert

The entire reason for the notification system is to receive alerts when there is a status change. To understand how an alert is received requires first an understanding of how the notification system works when there is not an alert.

As detailed in earlier use cases, each user on the system has their own UCF. Each UCF grants access to the ASF's and MSF's for the systems that the user is monitoring and, as applicable, can update. Each of these ASF's and MSF's contains a list of dependencies, that is other assets or missions that are being monitored. The health of those dependencies contributes to the overall health of the ASF or MSF in question.

When the user starts their UA, their UCF is loaded from the LNS. At this point, a sequence of activities occurs for each of the ASF's and MSF's. First, the status of ASF or MSF is compared to the last known status. Good status hygiene will dictate that part of the status contains the time at which the last change was made. This will ensure that if there were a series of changes that eventually ended in resolution and mission or asset returning to FMC that the downstream users be made aware there was a change to the status.

If last known status is different from the current status, then an alert is provided to the user through the UA that there has been a change to the status and the identification of the user who made the change.

This may have been done by a co-worker, that is an individual that works within the same work center for hierarchical missions or another individual who has received commander's intent and works as a near co-equal in non-hierarchical missions. This update may have been done by an echelon user, this is the next person in the chain of command for this asset or mission and has the authority to update the status file. It could also have been done by an administrator, either as the LNS, the ILNS, or the TLNS, based on information seen at that administrator's level that may not be available to the stakeholder or echelon user. For assets, it is possible that this update could be done by an automatic monitor based on activity observed that meets a certain pattern or based on a certain set of rules.

Who made the change is important. But also important is what was the change and why. In an ideal world with infinite personnel resources available, each ASF or MSF would have a stakeholder monitoring the item in question at all times. When a user starts their UA, loads the UCF, and starts to retrieve the statuses, it would be done while somebody else was also monitoring the ASF's and MSF's in question and communication between the person watching the item in question and the user just logging in could provide a shared Level 1 or Level 2 EMSA of the situation.

Situations will exist where there is not UA coverage for a mission at all hours of the day and night. Anything from inadequate manpower or importance to justify around the clock coverage to situations where all stakeholders are absent at a given time can cause a situation where it is not possible to immediately determine with person-to-person contact why a status file has changed. This is why the ASF's and MSF's must have a logging feature that will detail what the status changes were and why they were made to provide at least minimal Level 1 EMSA to personnel starting up their UA and seeing changes.

Once the ASF's and MSF's for which the user is responsible is queried for status changes and appropriate levels of EMSA are achieved, then each of the dependencies for the ASF's and MSF's are also checked for their statuses. Where there is a difference between the last recorded status for the dependency and the current recorded status, there is a reasonable assumption that there has been a change. An alert is provided in the UA that a dependency for one of the monitoring status files has changed, the difference between the last known status and the current status, and the number of changes that have occurred since the timestamp of the last known status. Figure 35 provides a flowchart example of this process.

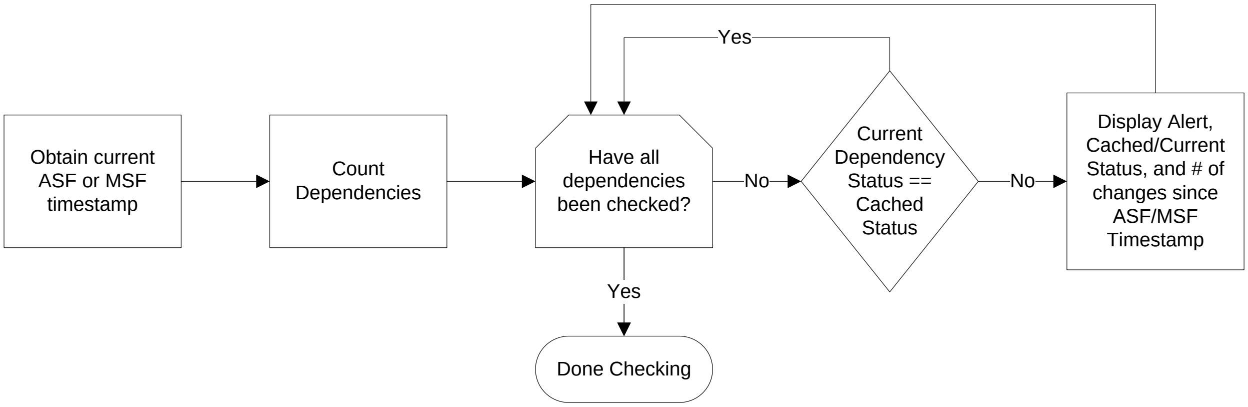

Obtain current ASF or MSF timestamp
Count Dependencies
Have all dependencies been checked?
No
Current Dependency Status == Cached Status
No
Display Alert, Cached/Current Status, and # of changes since ASF/MSF Timestamp
Yes
Yes
Done Checking

Figure 35. Flow Diagram for UA startup and UCF load

The number of alerts is critical because it is possible that the new status and the old status may say nearly the same thing and on the surface it would appear that there was no significant change. But if there were a significant number of changes between the current status and the last cached status, it is possible that there were significant problems that occurred in the intermediate alerts. It is possible, especially in cyber-related assets or missions that are heavily dependent on cyber-related assets, that the intermediate alerts tell a much bigger story and that these intermediate alerts might have some degree of consequence to those who rely on the mission even if the current status quo seems to demonstrate that there are no issues.

There are a number of issues that are not explored here as they are more implementation-based rather than theory based and as such fall outside of the scope of this thesis. One glaring issue deals with how many status changes to make available at any given time. Each status change is going to take up space. The more space that is allocated and utilized, the heavier the burden will be on the network infrastructure to move these updates. And as the number of available past alerts increases, so too does the concern that being able to see that many past alerts will create an OPSEC issue.

Once a baseline is established as to what the current status is of the assets and missions that a particular user is responsible for and the dependencies thereof, then the UA enters a normal operations mode. The normal operations mode and the start-up mode are very similar. The UA still checks the timestamp of the MSF or ASF for which they are a stakeholder or monitor, and do so on an interval commensurate with any number of factors. This includes importance of the mission or asset, the number of people who have stakeholder rights for the MSF or ASF, and the geographical distance between the stakeholders.

As a somewhat relevant example, a situation where two stakeholders are sitting in desks in the same small room and are responsible for a relatively minor mission may only check the status file time stamp every ten minutes whereas a situation where six stakeholders are sitting in different buildings scattered throughout a large installation and are responsible for a very important asset may check the status file time stamp every 30 seconds. The key is that the frequency of checks increases when it is more difficult to maintain a shared awareness of the status of the mission or asset and/or the importance of the mission or asset.

In a separate set of threads, the status of each of the dependencies is also checked on a time interval commensurate with the importance of the dependency to the overall health of the stakeholder's mission or asset. If the current status and cached status match, the thread goes inactive until the interval for checking is reached again. If the time stamps do not match, then an alert and the new status is presented in the UA.

There are a couple of points that are important to highlight here. First, the alert is presented to facilitate a shared awareness and/or Level 1 EMSA—the stakeholder(s) still have to decide what that change in status means to their mission or asset. This is where certain things can help in the decision making process (e.g. case-based reasoning or rules-based mission capability models.), but it is still the responsibility of the stakeholders to make that decision and publish it as required.

Second, and unique for this use case, is not all stakeholders will see the alert at the same time. There is no attempt in this system to synchronize the clocks of the various stakeholders, nor by definition is there necessarily a requirement to do so. If it has been identified that a particular dependency needs to be looked at every *n* seconds, it means that the stakeholders have determined that when the incident occurs or in *n* seconds after the incident occurs does not matter—as long as the notification is received during that threshold then the notification has been received in a timely manner as defined by the checking interval. Quicker notifications may be beneficial, but at the same time a prompt to grab a notification out of cycle may also provide insight to an attacker who is monitoring traffic to see if there is an increase in network traffic that coincides with the discovery of a status change.

With all stakeholders who are in the UA having a shared awareness or Level 1 EMSA on the alert the dependency has provided to them, they must then make a decision on what to do next. If there is no discernable effect to the mission or asset for which they are stakeholders, then doing nothing is a reasonable answer. If instead there is an impact to their ability to provide the mission or asset, then one of the stakeholders generates an update as detailed in use case #5 and publishes it to the LNS. As with the pulling of the dependency status, the stakeholders do not receive the updated status for that which they have authority for in their UA's until the next pull from the LNS.

One strength of this particular system is that if later another stakeholder connects to the LNS with his or her UA and sees a dependency alert that was not acted on by the other stakeholders but requires an update, then the stakeholder who recognizes the problem and is able to provide the appropriate Level 2 EMSA can make the update. Unlike a phone notification or a directed e-mail notification at one person, the dependency status changes live on until all stakeholders have a chance to see them, decreasing the chance that inadvertent filtering will occur.

As noted before in Use Case #5, if a LNS, ILNS, or TLNS administrator has set the AdminOverride variable for a cyber asset to true, the stakeholder of the cyber asset will not be able to update the ASF for that asset. Any items that are downstream from that asset and are within the control of that stakeholder may and should be updated if the incident on the cyber asset is responsible for any downstream effects that the stakeholder has responsibility for. It is through the admin's actions that the stakeholder will see that an admin has made an update in the UA.

Figure 36 illustrates the most complex of the use case sequence diagrams. Note that the loop continues until the UA is shut down. Figure 37 and 38 show the communication and activity diagrams respectively. Figure 37 is also very complex, but takes into account the different ways the notification system would handle missions versus assets and cyber assets verus non-cyber assets. Figure 38, like Figure 36 shows a continuous loop until such time that the UA is turned off.

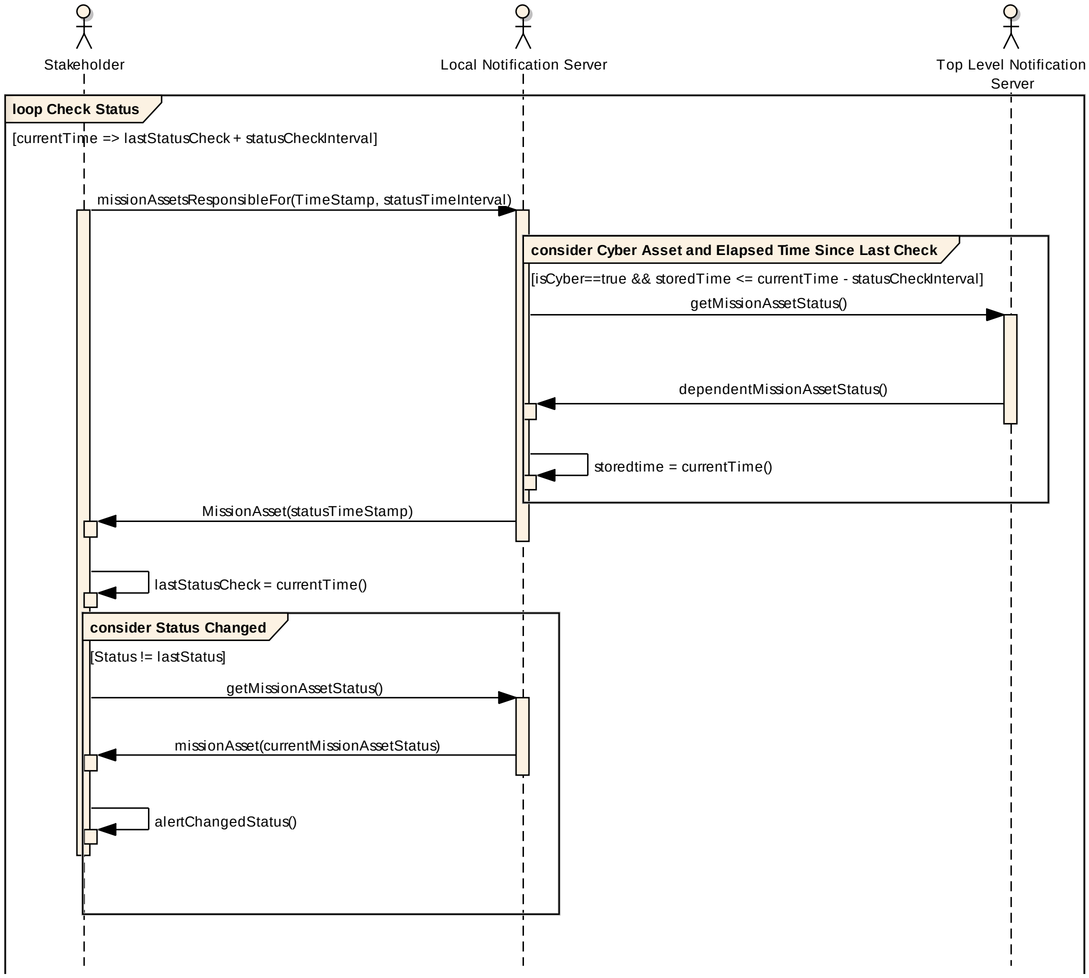

Stakeholder
Local Notification Server
Top Level Notification Server
loop Check Status
[currentTime => lastStatusCheck + statusCheckInterval]
missionAssetsResponsibleFor(TimeStamp, statusTimeInterval)
consider Cyber Asset and Elapsed Time Since Last Check
[isCyber==true && storedTime <= currentTime - statusCheckInterval]
getMissionAssetStatus()
dependentMissionAssetStatus()
storedtime = currentTime()
MissionAsset(statusTimeStamp)
lastStatusCheck = currentTime()
consider Status Changed
[Status != lastStatus]
getMissionAssetStatus()
missionAsset(currentMissionAssetStatus)
alertChangedStatus()

Figure 36. Sequence Diagram - Monitor Status File Receives Alert

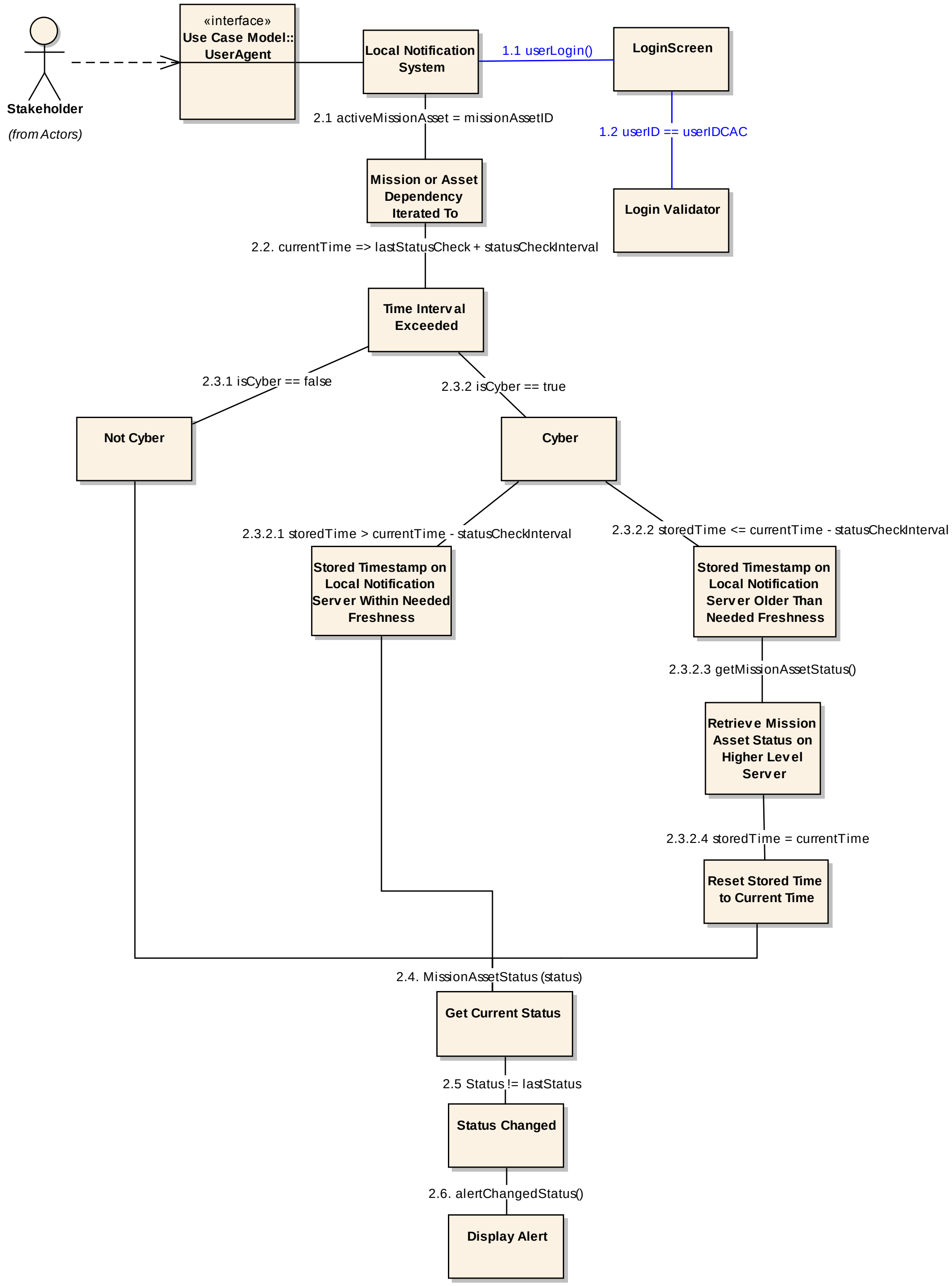


Figure 37. Communication Diagram - Monitor Status File Receives Alert

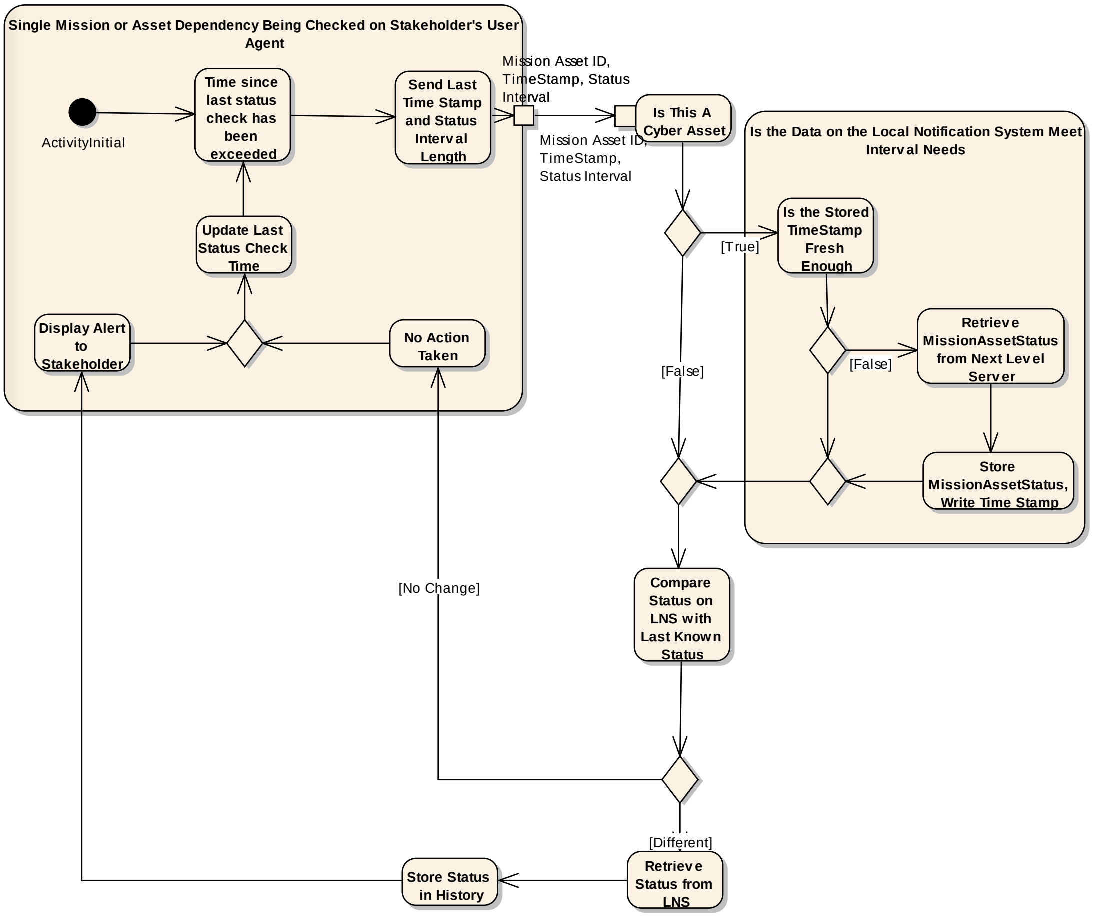

Single Mission or Asset Dependency Being Checked on Stakeholder's User Agent
ActivityInitial
Time since last status check has been exceeded
Send Last Time Stamp and Status Interval Length
Update Last Status Check Time
Display Alert to Stakeholder
No Action Taken
Mission Asset ID, TimeStamp, Status Interval
Mission Asset ID, TimeStamp, Status Interval
Is This A Cyber Asset
Is the Data on the Local Notification System Meet Interval Needs
[True]
Is the Stored TimeStamp Fresh Enough
[False]
[False]
Retrieve MissionAssetStatus from Next Level Server
Store MissionAssetStatus, Write Time Stamp
[No Change]
Compare Status on LNS with Last Known Status
[Different]
Retrieve Status from LNS
Store Status in History

Figure 38. Activity Diagram - Monitor Status File Receives Alert

5.9 Conclusion

The purpose of this chapter was to illustrate structural and behavioral constructs for the proposed notification architecture. This started with an overall use case diagram and class diagrams to set the structural picture. With the structural diagrams in place, behavioral diagrams were then presented for all six uses cases proposed as the minimum necessary to meet the stated research goals. These diagrams consisted of sequence, communication, and activity diagrams.

VI – Case for Action

6.1 Introduction

The first five chapters of this report have discussed CIMIA, illustrated some of the shortcomings of the manual notification systems, and demonstrated at a high level how a notification system that eschews push for pull would improve the process. This chapter will briefly review some of those themes prior to making the case to implement an architecture that connects stakeholders of missions and assets with those missions and assets that depend on them for mission accomplishment.

6.2 Current Notifications Methods are Insufficient

As discussed in the literature cited in previous chapters, the current methods to notify immediate downstream users fail in making timely and relevant notifications. Two primary schools of thought exist when it comes to making incident notifications.

The first, as outlined in AFI 33-138 [2], is for system owners to follow a hierarchical chain up to the top of network enterprise and for those administrators to then push the notifications back down the chain to users. Under the best of circumstances there was little confidence that all those who directly used an impacted system would get the notification based on a lack of *a priori* knowledge by network administrators as to who used what systems and at what level of importance. The significant reduction in manpower precipitated by PBD 720 [26], the resultant shift of traditional communications personnel from base level units to higher echelons in the enterprise, and the drawing back of WM's from operational units into the base communications structure [25] [28], has transformed a difficult notification scheme into an impossible one.

The second, though not discussed directly in the previous chapters, is where system owners are responsible for keeping track of who uses their systems and are responsible for pushing notifications to them when an incident occurs. Though not directly communicated in the text, the dynamic nature of the military makes this scheme impractical at best. This would require a system owner to engage in a continual process of tracking each user with a level of granularity that included alternate means and/or personnel to contact in case of an incident, how important the resource was to that user, and how quickly the downstream users needed to be contacted if an incident occurred. Such a requirement would impose an incredible burden in terms of time and resources, but would be theoretically possible if sufficient manpower existed. However, in the recent past of the United States military where the unofficial mantra has been "Do more with less," [51] or now "Do more with no more," [52] there is no possible means of maintaining such a system.

The primary means for receiving incident notifications have shown themselves to be systemically broken. Commanders ultimately have the responsibility for the missions assigned to them. Commanders delegate their authority for those missions to those that work for them [18]. These mission stakeholders have a vested interest in maintaining Level 1 EMSA [35] of the things they rely on for mission accomplishment. Rather than providing mission stakeholders with this needed degree of SA, instead the current notification system(s) provide little more than a self generated "fog of war." [12] We have to do better than this.

6.3 Threat to Our Networks Continues to Grow

Libicki [12] went into great detail about how the increasing size and scope of our networks provides for more opportunities for our enemies to attack us. Indeed, every indication from what is published in today's media is that network attacks continue to rise at an alarming rate. As such, it is reasonable to assume that failures in the CIA triad will continue to occur. Libicki also discussed that the threat exists from both outside and inside. Even the strongest defenses will not necessarily protect us from all malicious incidents when some of the threat exists inside the fortress. Nor will it protect us from well-meaning yet internally destructive information overload [53]. In the words of the late Walt Kelly, "We have met the enemy, and he is us." [54]

If even all of our defenses were perfect and everybody inside our castle walls behaved themselves, there also exists the threat of things breaking just because it was time for them to break. Microsoft issues patches nearly every Tuesday for their supported operating systems, showing that even (or especially) commercial operating systems continue to have flaws well after the time that they are released. And anything that relies on motion, electricity, or structural integrity is going to fail at some point, to include the structures in which are systems reside. Software or hardware is going to break at some point, leading to a computer incident for which a notification is going to have to be made. The aforementioned concepts of doing more with either less or no more will almost certainly include additional automation, thus adding more systems and creating more chance for structural, mechanical, or software failure.

6.4 Push is Systemically Broken for Incident Notifications

For reasons mentioned in the previous chapters and reiterated in 6.2 above, the level of accuracy of push notifications is insufficient for military operations when lives and national sovereignty is involved. No amount of metaphorical super glue, 100 mile per hour tape, bailing wire, and bubble gum is available to fix the problems that are part of push notification's DNA in an environment where the number of systems are increasing and the number of personnel to maintain them is decreasing.

Push also does not provide a reliable conduit for notifications for the downstream dependent missions that extend past the initial notification as defined in AFI 33-138 [2]. The instruction is silent to pushing this information beyond the bounds of those who own the systems and those who use the systems. But as mentioned in the discussion of missions, no mission sits alone. Almost certainly an incident on a cyber-based system is going to create impacts both to the mission that directly uses it but then also to the downstream users from that affected mission. The potential for second, third, and higher order effects from the cyber incident are almost guaranteed. An architecture to provide cyber notifications will be more beneficial for both full enumeration of mission impact and an enhancement of EMSA if these higher order effects can be communicated [35].

6.5 Mission Stakeholder Empowerment

As mentioned above, stakeholders have a vested interest in mission accomplishment. Part of this interest comes from pride. Part of this interest comes from a desire to be promoted. And part of this originates from avoiding the negative consequences associated with violating Article 92 of the Uniform Code of Military Justice, Dereliction of Duty [55].

Mission stakeholders rely on a variety of resources to remain capable of performing their given mission. As an example, a mission stakeholder for a mission that fixes electronic devices likely has a set of hand tools, spare parts, and repair manuals to perform that mission. That stakeholder can inventory the tools, parts, and manuals on a time interval consistent with how important these resources are to maintaining mission capability. Key to this is that he or she has physical control over these items and can choose to perform these checks as often as they want.

If the repair manuals go from being a paper product to an electronic product stored on a system outside of this stakeholder's span of control, then the stakeholder has to rely both that the stakeholder for the electronic manuals is going to perform their duty diligently, and that the providing stakeholder will remember the contact information for the repair facility should the manuals become unavailable or corrupted. This will allow the repair facility to either take alternate steps to remain mission capable or warn those who might use their mission that their capacity to perform the mission has been reduced.

What was missing before and after the transition to the electronic manuals was there was no automatic means for the stakeholder of the repair facility to notify his or her customers when there was a condition that impacted the facility's mission capability. At best there was, once e-mail became commonplace, the ability to send a message either directly to his or her customers (if such a list existed) or a mass e-mail to all who had e-mail. Choosing one or the other would either almost certainly guarantee customers were missed or that such an e-mail would contribute to or complete with e-mail noise which may or may not be filtered out.

If an architecture is available where stakeholders can subscribe to the status of missions, both in the cyber and physical domains, then there is no worry that a mission stakeholder will be forgotten when an issue arises that requires notification. While this empowerment sets up the line of communication, it does not directly solve the problem of the providing mission (in this case the owner of the system that stores the repair manuals) communicating the fact that there is a problem. An automatic monitoring system has the potential to help here, but short of that the burden of putting the information out to the world remains with the owner of the affected system. Inspiring the affected owner to publish the information has roots in various social science disciplines and falls well outside of the scope of this research.

This architecture also provides a conduit for the repair shop's mission stakeholder to then broadcast to his or her downstream users that there is an issue with the shop's ability to function. This is based again on converting the Level 1 EMSA received through the notification system, mixing it with the other factors that are being monitored, and then converting Level 1 to Level 2 or 3 EMSA for publishing to the notification system [35]. Then those missions who have identified the repair shop as a mission dependency will be able to receive that notification in a time frame commensurate with their need and the cycle begins anew.

At the same time the so called echelon users of the repair shop, that is the hierarchical levels up to and including the commander, can also see that there is an impact to the repair shop's mission and they also may assess their mission capability in light of the status update.

Overall this is a win-win-win-win for all involved. The winners include:

- The owner of the system responsible for electronic manuals wins. One notification is received by all of their echelon users, all of their dependent users, and AFNETOPS personnel at the same time. They can go to the business of either repairing that which is broken or working with network personnel to affect the fix.
- The stakeholder(s) for the mission(s) dependent on the electronic manual system wins. They have an immediate update that there is a problem so there is no unpleasant surprise when there is a reason to use the system and it is not available. They can assess what it means to their mission and mission capability and communicate that in the same way the owner of the electronic manual system could, starting that part of the process anew.
- Local network operations personnel win. The personnel at the base where the electronic manual system resides win because their efforts can now be concentrated on fixing the problem at hand without trying to figure out through speculative math how many missions were impacted and to what level of significance. Putting mission impact in the hands of those who know the mission best speeds up the repair process, resulting in the hastening of repairs for downstream users and reducing the impact to the missions involved.

- AFNETOPS personnel also win. They get a notification quicker because the status of cyber missions reside on their server. The opportunity exists for data mining, trend analysis, and overall enterprise-level health assessments by having all cyber statuses in one location.

6.6 Anatomy of a Cyber Incident – Current Methods versus Proposed Architecture

The following scenario is fictional and any resemblance to this and any real-world incident is purely coincidental. In this instance concentration is made on functional responsibility rather than the current naming conventions, so new names abound rather than that which is found either in AFI 33-138 or any transformation that has occurred since its publication. This scenario is borrowed from work on a not-yet published conference paper.

In a hypothetical military organization at Main Operating Base Alpha (MOB-A), a power supply fails in a server operating a single database. The database serves a small subset of users at a number of military bases, to include an organization at Main Operating Base Bravo (MOB-B). The owner of the database recognizes the problem.

In a traditional push communication method, the database owner notifies the local communications entity, the MOB-A Base Communication Center (BCC). The BCC pushes the message up to the next higher level of communications responsibility. The message is pushed up the chain twice more until it reaches the top level of communications responsibility (TLCR).

The TLCR is aware that the database could be used by other organizations but they do not know exactly who uses it. So the TLCR pushes a message to all of their immediate subordinate communications organizations. This push continues utilizing one push at a time down to all of the BCC's.

Provided that the MOB-B BCC gets the notification, a human there has to decide who to send the message to. As this is a reasonably specialized system, a message sent to all MOB B users will be noise. If the human instead decides to be selective as to whom they send it to, whether or not the dependent mission gets the notification will be purely on how much *a priori* knowledge that person has or how well they guess. If the guess was accurate and the message is received by the mission stakeholder, they can then assess how it will affect their mission. It is up to the mission stakeholder to then alert their downstream users with similar a priori knowledge of who might be affected by the resultant change in mission capability.

If instead the proposed architecture is used, the database owner uses their agent and publishes the status change. The local level server recognizes that it is a cyber asset and publishes the status change to the top level server.

Minutes later at MOB-B, a user agent requests the status of the cyber asset as it does periodically throughout the day. The local level server at MOB-B sees that the information it has stored is too old for the user need and requests a status update from the top level. The top level server responds with the most recent status which the local server echoes to the requestor and stores. Because this was a change in status, the user agent displays an alert.

With receipt and observation of the alert (Level 1 EMSA), the stakeholders responsible for the enduring mission that requested the update need to decide what this means to them (Level 2 EMSA) as well as make any changes to how they are performing their mission to accommodate for the lost resource (Level 3 EMSA) [35]. If it will negatively affect their ability to perform their mission, they publish that update to the local server and that information propagates to downstream missions dependent on that particular enduring mission when their user agents request an update. Because the mission that used the database at MOB-B was not related to a cyber asset, the update would go to the local level server but not the top level server.

Alternately, if this had been a compromise of data integrity or confidentiality, an appropriate warning would go out as well. This could warn downstream users that there could be problems with previously provided products and appropriate actions need to be taken. In the aforementioned scenario of a convoy database confidentiality breach, an alerting system such as this could save lives. Assisting the users with determining if/how the compromised information was used is a separate problem beyond the scope of this research.

6.7 Conclusion

The current methods of notification as defined in AFI 33-138 do not work as intended. It had a better chance of succeeding when first written, but organizational changes to the Air Force have rendered the scheme for notification to nothing more than words on a paper with no expectation of meeting timely and relevant notification. Restated here is the the case that push is a losing concept, both for initial notifications but also the lower level notifications to downstream dependencies. Explained in some detail is how stakeholder empowerment through subscription to a pull-type system aids in achieving Level 1 EMSA and is a winning proposition for all involved in the notification process. Also presented is a theoretical scenario where push notifications are compared to pull notifications.

VII – Conclusions and Recommendations

7.1. Conclusions

This research set out to help solve the problem of how to get timely and relevant notifications to mission stakeholders in response to a cyber incident. This includes not only the direct users of the system that were involved in the incident, but also those downstream users who rely on products that may have been tainted by the results of that incident.

In doing so, this research followed a long and somewhat winding path to, in, around, and sometimes circling the multitude of disciplines that CIMIA encompasses. This document looked briefly at workflow modeling before moving over to the precepts of UML and OOP. Situational awareness was explored to some depth and all who work to develop solutions for CIMIA have an enormous debt of gratitude to repay to Endsley for her research on this very abstract concept--it is the foundation that much of this particular architecture research rests upon. The intricacies of mission, mission impact, and mission capability were explored as well for it is one of the terms and concepts that creates the most consternation when trying to formulate solutions to this enormously complex research area. Communication theory, to include noise, push, and pull were explored at a depth commensurate with explaining the basic concepts that influenced this research. Also mentioned, yet quickly run away from, are the other behavioral and social science-based issues that add a deep layer of complication in an all ready complicated subject.

Some basic conclusions spring forward from this research. First and foremost is that the focus on CIMIA is planted firmly in the middle of the acronym: Mission. Mission and mission capability is why we as a military exist. Failure of the mission can have dire consequences to life, limb, property, national pride, and national sovereignty. All conversations about CIMIA should begin and end with mission.

Next is that the system for incident notifications, as written into the governing instruction, is systemically and hopelessly broken. The necessary and well-thought out decisions made to recover from the effects of PBD 720 diminished our ability to make timely and relevant notifications when cyber incidents occurred. The men and women working inside the Air Force's wing level communications squadrons have worked tirelessly to do the best job for their organizations and their customers. They are brothers and sisters at-arms. But metaphorically they have been playing solitaire with a deck of 51 cards [56] because the instruction governing cyber incident notifications did not mesh well with realities of communications theory and rapid growth in the usage of cyber-based assets for enduring missions.

Third is that creating the architecture for a notification system has to take into account both cyber incidents and non-cyber mission impacting events. Cyber, like space, is a domain that is all encompassing yet cannot work without the other domains. If the architecture proposed here is to be implemented, the use of the user agent should be encouraged for cyber and non-cyber events. An increase in Level 1 EMSA across all domains will benefit stakeholders and commanders at all levels and regular use of the UA will aid in familiarity when a cyber incident occurs.

7.2. Recommendations for Future Research

This particular research is but a small baby step towards an eventual answer or sets of answers for building a robust notification architecture for CIMIA. Undoubtedly this list could go on for many tens of pages. Three of highest promise are listed here.

The first avenue for additional research is trying to build a small-scale prototype of this concept. Building one to take into account a TLNS, LNS, and a small handful of missions would be the first step in either proving or disproving that such an architecture is feasible. Expanding the size and scope of the prototype to cover a couple of LNS entities and their associated users would move towards proving the scalability of such a system.

Another area to explore to take this concept further is expanding the depth to which the notification system looks at mission. One way would be incorporating data from other Air Force systems that also contribute to mission capability such as personnel, vehicles, and the like. A centralized place for a mission stakeholder to gain Level 1 EMSA on other aspects of their mission would extend the usefulness of the architecture and would further drive home the concept that cyber is but a part of the bigger process of maintaining mission capability.

And finally on that theme of mission capability, a strong argument could be made to combine this architecture with some more aspects of a decision support system. As explained in earlier chapters, rules-based mission capability has been used successfully in weapons systems and other applications. Solid introspection could build rules that would help mission stakeholders, especially when they are new to an assignment and are trying to learn the ways of doing business quickly and failure or impairment is not an option. Case-based reasoning could be of benefit here as well by looking at what the event is, comparing it with past events, and providing the stakeholder with options based on past successes and failures.

7.3. Final Thoughts

Technology does not replace the need for stakeholders to perform an adequate amount of introspection on the missions that they are responsible for. The best architecture, information, and plans will be foiled by a stakeholder who cannot extend Level 1 EMSA to Level 2 or Level 3 EMSA [35]. Leaders still have to be leaders.

7.4. Summary

This research presents the framework for a notification architecture to provide timely and relevant notification to mission stakeholders in response to the discovery of a cyber incident. The architecture then provides a means for the affected mission to share their status with those who depend on them and empower downstream users an opportunity to make similar decisions regarding their mission capability.

Chapter I presents background information regarding the subject, research, assumptions, and scope of research.

Chapter II presents a review of literature relevant to this area of study. This chapter covers a wide breadth of topics related to CIMIA. This includes breaking up the acronym CIMIA into the component parts and providing definition and clarity to them. Next is a discussion of what others have said are the biggest problems with getting timely and relevant notifications to users and this is expounded upon. SA and EMSA are discussed in how it relates to CIMIA. Workflow modeling is discussed briefly. And finally the difference between push and pull is explored.

Chapter III presents a set of investigative questions related to the research problem. The problem statement is broken into two main sections dealing with what is preventing notifications from being made and how the architecture that is proposed would break down some of those limitations.

Chapter IV revisits the different workflow models that were explored for this research, with UML ultimately being chosen.

Chapter V digs into the details of six use cases for how the proposed architecture would work. These explanations are buttressed with UML documents to provide a physical presence to a series of abstract concepts.

Chapter VI makes the case for implementing this architecture.

And this chapter sums up the research and offers vectors for others to travel on to eventually solve the ultimate problem: Getting timely and relevant notifications to mission stakeholders so that they may assess the mission impact of incidents.

Appendix A: Acronyms Used

Acronym	Definition
AB	Air Base
AFCYBER	Air Force Cyber Command
AFDD	Air Force Doctrine Document
AFI	Air Force Instruction
AFNETOPS	Air Force Network Operations
AFNOSC	Air Force Network Operations and Security Center
AFPAM	Air Force Pamphlet
AFSC	Air Force Specialty Code
AOR	Area of Responsibility
ASF	Asset Status Files
ASI	Authorized Service Interruption
BDA	Battle Damage Assessment
C2RMS	Command and Control Remote Monitoring System
C4	Command, Control, Communications, and Computer
CAC	Common Access Card
CFP	Communications Focal Point
CIA	Confidentiality, Integrity, and Availability
CIMIA	Cyber Incident Mission Impact Assessment
CLearn	Commander's Learning Agent
DDOS	Distributed Denial of Service
DNS	Domain Name Server
DoD	Department of Defense
DoDAF	Department of Defense Architecture Framework
DOS	Denial of Service
EET	Exercise Evaluation Team
EMSA	Endsley Model Situational Awareness
FMC	Full (or Fully) Mission Capable
HAF	Headquarters Air Force
IDEF	Integration Definition
ILNS	Intermediate Notification System
JP	Joint Publication
LNS	Local Notification System
LNSA	Location Notification System Administrator
MAIN	Mission/Asset Identification Number
MAJCOM	Major Command
MEF	Mission Essential Functions
MSAA	Mission Service Automation Architecture
MSF	Mission Status Files

NAF	Numbered Air Force
NCC	Network Communications Center
NCOIC	Non-Commissioned Officer In Charge
NMC	Not (or Non) Mission Capable
NOSC	Network Operations and Security Center
NOTAM	Notice to Airmen
OOP	Object-Oriented Programming
OPSEC	Operations Security
PBD	Programmed Budget Decision
PMC	Partial (or Partially) Mission Capable
RSS	Really Simple Syndication
SA	Situational Awareness
TLNS	Top Level Notification System
UA	User Agent
UCF	User Configuration Files
UML	Unified Modeling Language
USI	Unscheduled Service Interruption
WM	Workgroup Manager
XML	Extensible Markup Language
YAWL	Yet Another Workflow Language

Bibliography

[1] D. Sorrels, M. Grimaila, L. Fortson, and R. Mills, "An Architecture for Cyber Incident Mission Impact Assessment (CIMIA)," in *International Conference on Information Warfare and Security (ICIW 2008)*, University of Nebraska, 2008.

[2] Department of the Air Force, *Air Force Instruction 33-138: Enterprise Network Operations Notification and Tracking*. Washington, DC, 2005.

[3] Department of Defense, *Joint Publication 1-02: Department of Defense Dictionary of Military and Associated Terms*. Washington, DC, 2009.

[4] M. Linderman et al., "A Reference Model for Information Management to Support Coalition Information Sharing Needs," in *Tenth International Command and Control Technology Symposium (ICCRTS)*, Tysons Corner, 2005.

[5] S. Musman, A. Temin, M. Tanner, D. Fox, and B. Pridemore, "Evaluating the Impact of Cyber Attacks on Missions," in *Proceedings of the 5th International Conference on Information Warfare and Security*, Wright-Patterson AFB, OH, 2009, pp. 446-456.

[6] M. Grimaila, L. Fortson, and J. Sutton, "Design Considerations for a Cyber Incident Mission Impact Assessment (CIMIA) Process," in *International Conference on Security and Management (SAM 09)*, Las Vegas, 2009.

[7] Department of the Air Force, *Air Force Doctrine Document 3-12: Cyberspace Operations*. Washington, DC, 2010.

[8] A. Bargar, "DoD Global Information Grid Mission Assurance," *CrossTalk: The Journal of Defense Software Engineering*, July 2008.

[9] M. Bishop, *Computer Security: Art and Science*. Boston: Addison-Wesley, 2003.

[10] D. Bell and L. La Padula, "Secure computer system: Unified exposition and Multics interpretation," Mitre Corp, Bedford, 1976.

[11] K. Biba, "Integrity Considerations for Secure Computer Systems," Electronic Systems Division, 1977.

[12] M. Libicki, *Conquest in Cyberspace*. New York: Cambridge, 2007.

[13] D. Pipkin, *Information Security: Protecting the Global Enterprise*. Upper Saddle River: Prentice Hall, 2000.

[14] F. Lau, S. Rubin, and M. Smith, "Distributed denial of service attacks," in *IEEE Conference on Systems, Man, and Cybernetics*, Nashville, 2000, pp. 2285-2280.

[15] Department of the Air Force, *Air Force Pamphlet 63-128: Guide to Acquisition and Sustainment Life Cycle Management*. Washington, DC, 2009.

[16] Iceland Radar Agency, *Iceland Air Defense System Maintenance Control Operating Instruction*, 2001.

[17] Department of the Air Force, *Air Force Doctrine Document 1-1: Leadership and Force Development*. Washington, DC, 2006.

[18] Department of the Army, *Army Regulation 600-20: Army Command Policy*. Washington, DC, 2010.

[19] Department of the Air Force, *Air Force Instruction 38-101: Manpower and Organization*. Washington, DC, 2006.

[20] Department of the Air Force, *Air Force Instruction 10-2501: Air Force Emergency Management (EM) Program Planning and Operations*. Washington, DC, 2009.

[21] M. Grimaila and L. Fortson, "Improving the Cyber Incident Damage and Mission Impact Assessment," *IAnewsletter*, vol. 11, no. 1, Spring 2008.

[22] M. Grimaila, G. Schechtman, and R. Mills, "Improving Cyber Incident Notification in Military Operations," in *Proceedings of the Institute of Industrial Engineers Annual Conference*, Miami, 2009.

[23] J. Goodall, A. D'Amico, and J. Kopylec, "Camus: Automatically Mapping Cyber Assets to Missions and Users," in *IEEE Military Communications Conference*, Boston, 2009.

[24] J. Stanley, R. Mills, R. Raines, and R. Baldwin, "Correlating Network Services with Operational Mission Impact," in *Military Communications Conference (MILCOM 2005)*, Atlantic City, 2005.

[25] W. Lord, "Revised Standard Base-Level Communications Structure", 2007.

[26] Air Force Audit Agency, "Audit Report: Air Force Personnel Reductions - FD2008-0004-FD4000," 2008.

[27] J. Donnithorne, "Tinted Blue: Air Force Culture and American Civil-Military Relations," *Strategic Studies Quarterly*, vol. 4, no. 4, Winter 2010.

[28] J. Nelson, "3DXXX Enlisted AFSC Conversions", 2009.

[29] B. Hale, M. Grimaila, R. Mills, M. Haas, and P. Maynard, "Communicating Potential Mission Impact Using Shard Mission Representations," in *Proceedings of the 5th International Conference on Information Warfare and Security*, Dayton, 2010.

[30] L. Tinnel, O. Saydjari, and J. Haines, "An Integrated Cyber Panel System," in *Proceedings of the 2003 DARPA Information Survivability Conference and Exposition*, Washington, DC, 2003, pp. 32-34.

[31] D. Alberts and A. Hayes, *Power to the Edge: Command. Control. in the Information Age*. Washington, DC: DoD Command and Control Research Program, 2003.

[32] B. Jos and T. Culbertson, "Leveraging Net-Centric Monitoring Techniques with Information Fusion to Increase US Air Force Information Dominance," in *Military Communications Conference (MILCOM 2006)*, Washington, DC, 2006.

[33] D. Alberts, J. Garstka, and F. Stein, *Network Centric Warfare: Developing and Leveraging Information Superiority*. Washington, DC: DOD Command and Control Research Center Publications, 1999.

[34] D. Alberts, J. Garstka, R. Hayes, and D. Signori, *Understanding Information Age Warfare*. Washington DC: DOD Command and Control Research Center Publications, 2001.

[35] M. Endsley, "Toward a Theory of Situation Awareness in Dynamic Systems," *Human Factors*, vol. 37, no. 1, pp. 32-64, 1995.

[36] Department of Defense. (2011, Jan.) DoDAF Architecture Framework Version 2.02. [Online]. http://cio-nii.defense.gov/sites/dodaf20/

[37] D. Georgakopoulos, M. Hornick, and A. Sheth, "An overview of workflow management," *An overview of workflow management: From process modeling to workflow automation infrastructure*, vol. 3, no. 2, pp. 119-153, 1995.

[38] W. Van Der Aalst and A. Ter Hofstede, "YAWL: yet another workflow language ," *Information Systems*, vol. 30, no. 4, pp. 245-275, 2005.

[39] C. Menzel and R. Mayer, Handbook on architectures of information systems, 2006.

[40] C. Kim, D. Yim, and R. Weston, "An integrated use of IDEFO, IDEF3 and Petri net methods in support of business process modelling," *Journal of Process Mechanical Engineering*, vol. 215, no. 4, pp. 317-329, 2001.

[41] A. Dennis, B. Wixom, and D. Tegarden, *Systems Analysis & Design: An Object-Oriented Approach with UML.*: Wiley & Sons, 2001.

[42] J. Schmuller, *Teach Yourself UML in 24 Hours*, 3rd ed. Indianapolis: Sams, 2004.

[43] D. Winer. (2005) RSS 2.0 Specification. [Online]. http://www.rssboard.org/rss-specification

[44] M. Grimaila, L. Fortson, and J. Sutton, "Design Considerations for a Cyber Incident Mission Impact Assessment (CIMIA) Process," in *International Conference on Security and Management (SAM 09)*, Las Vegass, 2009.

[45] Air Force Personnel Center Public Affairs. (2010, August) Air Force to release staff sergeant promotion list. [Online]. http://www.afpc.randolph.af.mil/news/story.asp?id=123216937

[46] Department of the Air Force, *Air Force Instruction 36-2618: The Enlisted Force Structure*. Washington DC, 2009.

[47] Department of Defense, *DoD 8570.01-M: Information ASsurance Workforce Improvement Program*. Washington, DC, 2010.

[48] Deparment of the Air Force, *CFETP 3A0X1: AFSC 3A0X1 Knowledge Operations Management*. Washington, DC, 2008.

[49] Milcord. (2011, January) Commander's Learning Agent - MilcordWiki. [Online]. http://wiki.milcord.com/index.php/Commander's_Learning_Agent

[50] A. Shaw, "A Model For Performing Mission Impact," Air Force Institute of Technology, Wright-Patterson AFB, Ohio, Masters Thesis 2007.

[51] S. Thatcher. (2010, September) The Secret of Doing More With Less. [Online]. http://www.mcconnell.af.mil/news/story.asp?id=123214401

[52] A. Carter, Better Buying Power: Mandate for Restoring Affordability and Productivity in Defense Spending, 2010.

[53] P. Marksteiner, "The threat from within: E-mail overload degrades military decision-making," *Armed Forces Journal*, September 2008.

[54] (2011, January) I Go Pogo - We have met. [Online]. http://www.igopogo.com/we_have_met.htm

[55] Department of Defense, *Manual for Courts-Martial, United States*. Washington DC, 2010.

[56] L. DeWitt, Flowers on the Wall, 1965.

Vita

Biographical Sketch

Master Sergeant James L. Miller grew up in Paso Robles, California. He graduated from Paso Robles Joint Union High School in June 1989 and attended the University of La Verne in La Verne, California prior to enlisting in the Air Force under the Delayed Enlistment Program in January 1992. He entered active duty on 18 March 1992. He completed technical training as a Joint Surveillance System Electronic Computer and Switching Apprentice and has built a vast breadth of experience during his career. He has served at various levels of the Air Force hierarchy spanning operating location, flight, sector, and squadron.

Sergeant Miller is currently a student at the Air Force Institute of Technology (AFIT) pursuing a Master of Science Degree in Cyber Operations. Prior to attending AFIT, Sergeant Miller served as Section Chief, Communications Infrastructure, 17th Communications Squadron, 17th Mission Support Group, 17th Training Wing, Goodfellow AFB (AETC), Texas. In this duty he led over 40 Airmen in all aspects of the wing's telephone, computer network, and video teleconference infrastructure, serving over 3,500 permanently assigned Airmen and an additional 13,000 intelligence, fire fighting, and special instruments students each year. He also served as an additional duty First Sergeant, contributing to the morale and discipline of the over 150 assigned enlisted Airmen in the squadron and providing guidance to the squadron's commander and director.

Education

Master of Science, Cyber Operations, Air Force Institute of Technology, Wright-Patterson AFB, Ohio. Master's Thesis: An Architecture for Improving the Timeliness and Relevance of Cyber Incident Notifications And Their Dependent Missions Chair: Robert F. Mills, PhD. In progress. Expected graduation date: March 2011.

Bachelor of Science, Computer Studies, University of Maryland University College, Adelphi, Maryland, May 2005.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)* 24-03-2011	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From – To)* Mar 2010-Mar 2011

4. TITLE AND SUBTITLE An Architecture for Improving Timeliness and Relevance of Cyber Incident Notifications	**5a. CONTRACT NUMBER** 10ENV297
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Miller, James L., MSgt, USAF	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GCO/ENG/11-09
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Douglas Kelly, PhD, Cyber Team Lead Air Force Research Laboratory 711th Human Performance Wing Sense-making and Organizational Effectiveness Branch (RHXS) 2698 G Street, Bldg 190 Wright-Patterson AFB OH 45433-7604 Comm: (937) 656-4391	**10. SPONSOR/MONITOR'S ACRONYM(S)** AFRL/HPW/RHXS **11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States

14. ABSTRACT

This research proposes a communications architecture to deliver timely and relevant cyber incident notifications to dependent mission stakeholders. This architecture, modeled in Unified Modeling Language (UML), eschews the traditional method of pushing notifications via message as dictated in Air Force Instruction 33-138. It instead shifts to a "pull" or "publish and subscribe" method of making notifications. Shifting this paradigm improves the notification process by empowering mission owners to identify those resources on which they depend for mission accomplishment, provides a direct conduit between providing and dependent mission owners for notifications when an incident occurs, and provides a shared representation for all with authority for that dependent mission. Once the incident's impact is assessed, the architecture provides a conduit for the mission stakeholder(s) receiving the incident notification to then notify their downstream users of their status should it have changed because of the incident. The proposed architecture significantly speeds incident notification by eliminating multiple layers of processing and does so in a relatively noise-free environment as compared to current notification methods.

15. SUBJECT TERMS
CIMIA, Mission Impact Assessment, Incident Notification

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Robert F. Mills (ENG)
REPORT U	**ABSTRACT** U	**c. THIS PAGE** U	UU	156	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-3636 X4527, robert.mills@afit.edu

Standard Form 298 (Rev: 8-98)
Prescribed by ANSI Std. Z39-18